Verb & Adjectives.

1

United States Patent [19]

Paeseler

[11] **Patent Number:** **4,947,438**

[45] **Date of Patent:** **Aug. 7, 1990**

[54] PROCESS FOR THE RECOGNITION OF A CONTINUOUS FLOW OF SPOKEN WORDS

[75] Inventor: **Annedore Paeseler**, Hamburg, Fed. Rep. of Germany

[73] Assignee: **U.S. Philips Corporation**, New York, N.Y.

[21] Appl. No.: **217,535**

[22] Filed: **Jul. 11, 1988**

[30] **Foreign Application Priority Data**

Jul. 11, 1987 [DE] Fed. Rep. of Germany 3723078

[51] **Int. Cl.**^{5} **G10L 5/00**

[52] **U.S. Cl.** **381/43**; 364/513.5

[58] **Field of Search** 381/41–43; 364/513, 513.5

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,156,868	5/1979	Levinson	381/43
4,277,644	7/1981	Levinson et al.	381/43

FOREIGN PATENT DOCUMENTS

3215868 3/1982 Fed. Rep. of Germany .

OTHER PUBLICATIONS

Woods, "Transition Network Grammar for Natural Language Analysis", Communications of the ACM, vol. 13/No. 10/Oct., 1970, pp. 591–606.

Tennant, Natural Language Processing, Petrocelli book, New York/Princeton, 1981, pp. 75–101.

Niitsu et al., "A Method of Using Linguistic Information for Automatic Spoken Word Recognition", Systems.Computer.Controls, vol. 10, No. 1, 1979.

Tappert, "A Preliminary Investigation of Adaptive Control in the Interation Between Segmentation and Segment Classification "in Automatic.

"Recognition of Continuous Speech", IEEE Trans. on Systems, Man, and Cybernetics, vol. SMC-2, No. 1, Jan. 1972, pp. 66–72.

H. Ney, "Dynamic Programming Speech Recognition . . ." Proc. ICASSP, IEEE Conf., Dallas (4/87) pp. 69–72.

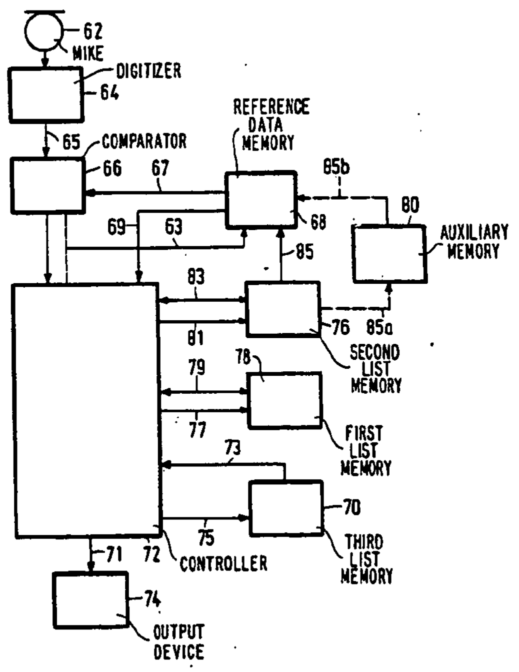
J. Earley, "An Efficient Context-Free Parsing Algorithm" comm. of the ACM, vol. 13, No. 2, pp. 94–102 (2/70).

Primary Examiner—Gary V. Harkcom
Assistant Examiner—David D. Knepper
Attorney, Agent, or Firm—Anne E. Barschall

[57] **ABSTRACT**

Continuous speech recognition assigns predetermined words to syntactic categories and defines the syntactic categories which can follow and precede each predetermined word. The recognition process is achieved by comparing the input sequence of speech signals to reference values and summing those which are syntactically permissible until they form a valid word. Subsequent speech values to previouly calculated valid words are compared to reference values listed in syntactic categories which can follow the predetermined word. For each word, values are updated indicating the current word's sequence number, syntax category, cumulative comparison sum, and the current list of compared words. Values are also stored for each word which identify the previous word, the following word and their syntax categories. This process is repeated until all input values have been processed. The results are then checked to verify valid syntax and the words with the closest match are read out.

13 Claims, 3 Drawing Sheets



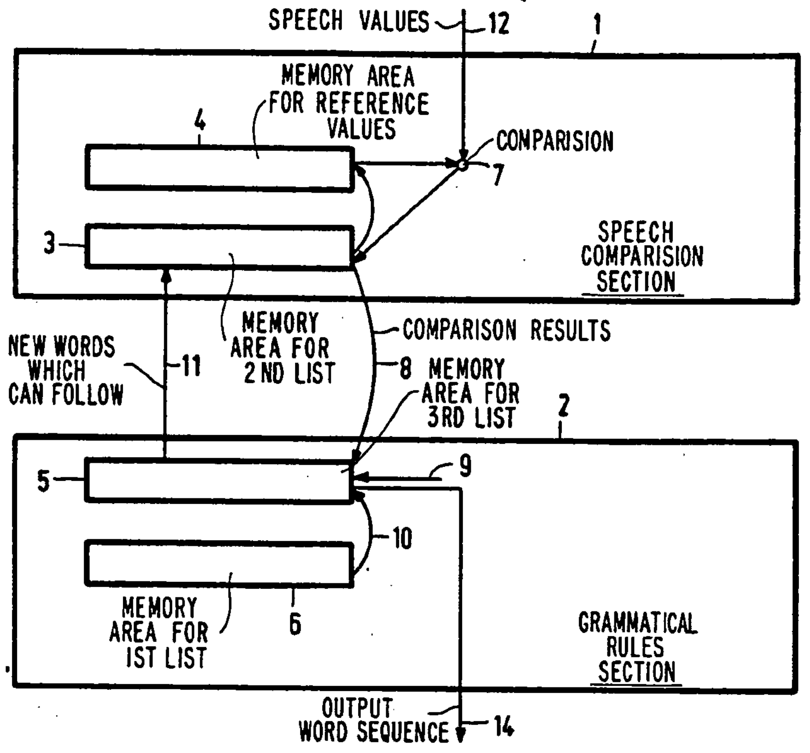

SPEECH VALUES
12
1
MEMORY AREA FOR REFERENCE VALUES
4
COMPARISION
7
3
SPEECH COMPARISION SECTION
COMPARISON RESULTS
NEW WORDS WHICH CAN FOLLOW
11
MEMORY AREA FOR 2ND LIST
8 MEMORY AREA FOR 3RD LIST
2
5
9
10
MEMORY AREA FOR IST LIST
6
GRAMMATICAL RULES SECTION
OUTPUT WORD SEQUENCE
14

FIG.1

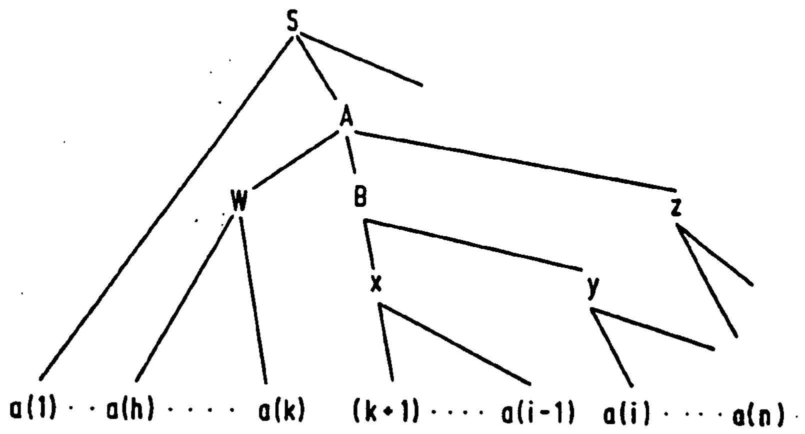

S
A
W
B
z
x
y
a(1) · · a(h) · · · · a(k) (k+1) · · · · a(i-1) a(i) · · · · a(n)

FIG. 4

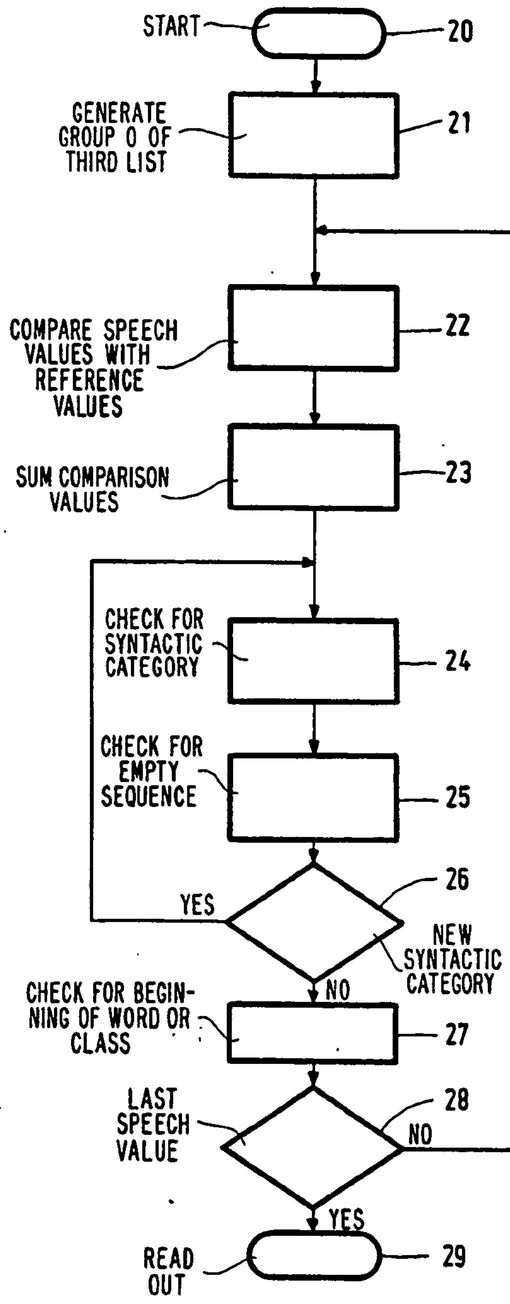

START
20
GENERATE GROUP 0 OF THIRD LIST
21
COMPARE SPEECH VALUES WITH REFERENCE VALUES
22
SUM COMPARISON VALUES
23
CHECK FOR SYNTACTIC CATEGORY
24
CHECK FOR EMPTY SEQUENCE
25
26
YES
NEW SYNTACTIC CATEGORY
NO
CHECK FOR BEGINNING OF WORD OR CLASS
27
LAST SPEECH VALUE
28
NO
YES
READ OUT
29

FIG. 2

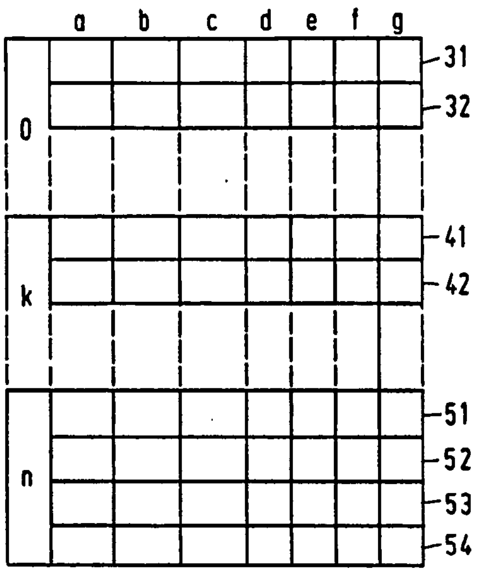

a
b
c
d
e
f
g
o
k
n
31
32
41
42
51
52
53
54

FIG.3

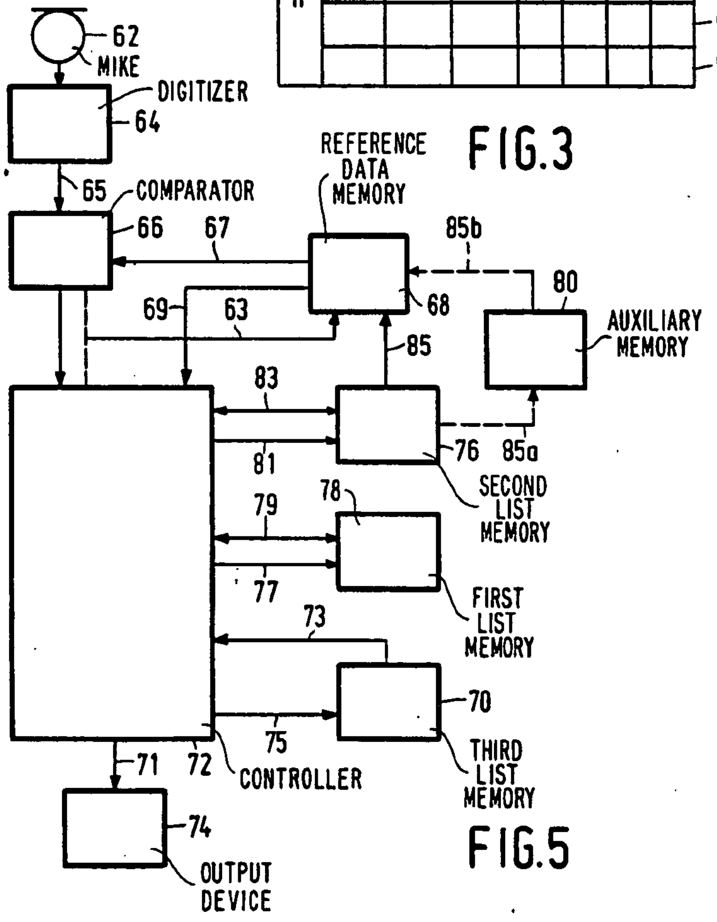

62
MIKE
DIGITIZER
64
65
COMPARATOR
66
67
REFERENCE DATA MEMORY
68
69
63
85b
80
AUXILIARY MEMORY
85
83
81
76
85a
SECOND LIST MEMORY
79
78
77
FIRST LIST MEMORY
73
70
75
THIRD LIST MEMORY
71
72
CONTROLLER
74
OUTPUT DEVICE

FIG.5

PROCESS FOR THE RECOGNITION OF A CONTINUOUS FLOW OF SPOKEN WORDS

The invention relates to a process for the recognition f a speech signal derived from a continuous flow of spoken word. The speech signal consists of a temporal sequence of speech values, each of which specifies a section of the speech signal. The speech values are compared with predetermined stored reference values. A group of the reference values in each case represents one word of a predetermined vocabulary. The comparison results are added up over various sequences of combinations of reference values and speech values per sequence. Only such sequences of words are taken into account whose order is permissible in accordance with a predetermined stored first list containing, for predetermined syntactic categories, at least one assignment per category to a combination of further syntactic categories and/or words.

A process of this kind is known from "Proc. ICASSP IEEE Conf. ASSP", Dallas, Apr. 1987, p. 69-72. In this known process, the first list is divided into two sublists, which on the one hand specify the assignment between words and predetermined syntactic categories and on the other hand specify the assignment of these categories to two other, subordinate categories where appropriate. Both lists are used for each new speech signal, in that retrospective observation is used each time to determine which category explains the preceding speech section best. At the end of the speech signal, that sequence of words can be traced back which resulted in the smallest total sum of all comparison results and which moreover correspond to the grammar provided by the two lists. However, as a result of the retropsective observation for each new speech signal, it may occur that a sequence is not directly traced back to the beginning and consequently the process eventually recognizes two or even more sub-sequences within the speech signal which are in each case grammatically correct within themselves but the sub-sequences do not match each other grammatically.

The object of the invention is therefore to state a process of the type mentioned at the beginning which functions more reliably and makes fewer demands of the form, that is to say the assignments of the first list, so that even more than two further syntactic categories and/or words can be assigned to a syntactic category.

This object is achieved according to the invention, in that a second list containing at least references to the references values of all those words which are compared with the respective next speech value as well as a sequence number per word, and in that in the course of the process, a third list is generated which contains, for each speech value which has been compared with the last reference value of at least one word, a group having in each case a plurality of entries, each entry containing in addition to a current sequence number

(a) a reference to a syntactic category of the first list,
(b) a first specification for a sequence of compared words and/or syntactic categores which are assigned to sequences of already compared speech values,
(c) a second specification for a sequence of words and/or syntactic categories which can be assigned to subsequent speech values on the basis of the first list,

(d) a further sequence number assigned to the respective entry,
(e) a first evaluation value,
(f) a second evaluation value and
(g) a sequence of compared words,

in that at least after every comparison of a new speech value with the last reference value of at least one word, a new sequence number is determined and after each such comparison the group of entries of the third list associated with the sequence number stored in the second list at this word is searched through for such entries in which the sequence contained in the second specification begins with the compared word and, for each such entry present, a new entry is derived for the new group of the third list associated with the new sequence number,

in that subsequently, for each new entry where the abbreviated sequence contained in the second specification begins with a syntactic category for which at least one assignment is present in the first list, first further entries are made in the new group and, furthermore, a second further entry for the new group is derived for each of the new and first further entries of the new group for which the second specification contains an empty sequence,

in that deriving and making the first and second further entries alternately is repeated until, after at least one first further entry, no second further entry occurs,

in that subsequently, for all entries of the new group where the second sequence begins with a word to be recognized, a reference to the reference data of this word is entered into the second list,

in that subsequently the next speech value is compared with the reference values of all words contained in the second list, and

in that this course of process steps is repeated until the last speech value of the speech signal to be recognized, after the processing of which the last group of the third list is checked for all entries containing a reference to the syntactic initial category and, as a second specification, an empty sequence and, as a sequence number, that of the first group, and, from these entries, the sequence of compared words is read out and output form that entry having the smallest first evaluation value.

The process according to the invention combines the advantages of a retrospectively directed hypothesization of grammatically correct continuations for the already processed part of the speech signal with a continuous verification of these hypotheses starting from the beginning. Moreover, the process according to the invention has the result that only those words, or the associated reference values, are compared with the input speech values that are permissible on the basis of the grammar defined in the first list. Sampling values of the speech signal which were obtained at 10 ms intervals and reduced to their spectral values can be used as speech values. Other measures for preparing the speech sampling signals may, also be used; similarly the speech values can be obtained from a plurality of sampling values and represent, for example, diphones or phonemes or even larger units, which makes no essential difference to the process according to the invention.

The process according to the invention has certain formal similarities with a process described in "Comm. of the ACM", Vol. 13, No. 2, Feb. 1970, p. 94-102. This process is, however, used to break down sentences in a written and hence unambiguous form into their grammatical components. This, however, presents problems in the automatic recognition of a continuous flow of speech, in that the manner of speaking and speed of speaking vary and the transitions between the words are fluid. The words within a sentence cannot therefore be determined reliably, but only with a certain degree of probability. During recognition, therefore, many different hypotheses for words or sequences of words must be considered. For an automatic recognition, then, that word sequence is to be determined which was spoken with the greatest probability.

It follows from the unreliability of the input data that for a larger vocabulary, for example of several hundred words, each new speech value input must be compared with a very large number of different reference values obtained from the combination of all words, the words occuring, due to the impossibility of exactly determining the word boundaries, with different lengths with respect to the reference words. When the restrictions which the grammar imposes on the combination possibilities of words are taken into account, the number of the words and hence of the reference values with which each new speech value must be compared can be restricted.

The speech values are expediently compared with the reference values according to the method of dynamic programming, which is known, for example, from German Offenlegungsschrift 3,215,868. It permits adaptation to a varying speed of speaking and the finding of the most probable word.

The process according to the invention is essentially defined by the use of the second list and the special structure of the third list. A particularly favourable order, particulary also with respect to the processing time, is obtained according to a development of the invention,

in that before the comparison of the first speech value, the first group contains in first entries in each case a reference to a syntactic initial category, an empty sequence as a first specification, in each case another of the combinations assigned to the initial category as a second specification, and an intial value for the two evaluation values, as well as, in further entries, all the categories which can be derived from the combinations of the second specification with in each case corresponding combination, in that each new entry in the first specification contains a sequence extended to include the compared word, a sequence abbreviated to exclude the compared word in the second specification, the evaluation value incremented by the sum of the comparison results of the word as a first evaluation value, the sequence extended to include the compared word as a sequence of compared words, and furthermore the values of the entry present,

in that the first further entries contain in each case a reference to the syntactic category of the new entry, from which this first further entry is derived, an empty sequence in the first specification, in each case another combination assigned to the syntactic category as a sequence in the second specification, the new sequence number as a further sequence number, the evaluation value of the new entry for both evaluation values, and an empty sequence as a sequence of compared words,

in that for each second further entry, from that group specified in the further sequence number of the relevant new or further first entry, that earlier entry is read out where the sequence associated with the

second specification begins with the syntactic category to which this new or first further entry of the new group contains a reference, wherein the second entry contains the reference t the syntactic category of the earlier entry, a sequence extended to include the syntactic category of the current new or first further entry in the first specification, a sequence abbreviated to exclude the same syntactic category in the second specification, the sequence number of the earlier entry as a further sequence number, the sum of the first evaluation value of the earlier entry and the difference between the two evaluation values of the current entry as a first evaluation value, the corresponding evaluation value of the earlier entry and the sequence of compared words of the earlier entry extended to include the sequence of compared words of the entry of the current group as a second evaluation value, and

in that each reference to the reference data, together with the associated first evaluation value and the sequence number of the relevant entry, is entered into the second list. In this way, the evaluations are taken into account particularly well when completing recognized word sequences and when hypothesizing about correct continuations.

In the process according to the invention, it is possible that in the course of recognition of sentence, two or even more hypotheses converge at a common point, that is to say two or more hypotheses result in the same grammatical continuation, with the evaluation of the converging hypotheses being different in general. In this case, it is expedient, according to a further development of the invention, that each first and second further entry is only made provided that no entry is present in the new group, which entry contains the same reference, the same first and second specification and the same further sequence number and in which the first evaluation value is smaller than the first evaluation value of the intended further entry and, in the case of such an entry being already present but with a greater evaluation value, the latter is deleted. Therefore, as a result of this, converging hypotheses are not separately traced further, but only the best one is taken into consideration further, since the other hypothese cannot obtain a better overall evaluation at the end of the sentence to be recognized due to the same continuation. The conditions checked here ensure that really only those hypotheses are combined which result in exactly the same continuation. This represent, therefore, a recombination of hypotheses on the grammatical level.

It is of course possible that, in a group of entries in the third list where there is a plurality of entries, the second sequence begins with the same word or the same method. In this case it is expedient according to a further development of the invention that a reference to reference data is only entered into the second list provided that no reference to the reference data of the same word and the same sequence number with a smaller evaluation value is already present in said list, and, in the case of such an entry beng already present but with a greater evaluation value, the latter is deleted. A word or a word class need only be transferred from a group into the second list once, since after the complete comparison of this word, all entries of the relevant group are searched through to find whether the second sequence begins with this word therein. When the word from the entry with the smallest evaluation value is transferred into the second list, this entry specifies the best possible evaluation of a word sequence which has led to the word transferred into the second list. On the other hand, the same word is transferred again and again into the second list from various groups.

In the case of a vocabulary range necessary for realistic application, however, the number of words to be simultaneously compared, reduced by taking account of the grammar, is still too large, so that to speed up the process, the search must be concentrated on the most promising hypotheses. This takes place according to a development of the invention in that each new and each first and second further entry is only made if its first evaluation value is smaller than a threshold value which is equal to the smallest evaluation value, extended by a constant, of all entries currently contained in the second list. Only such hypotheses are traced further, therefore, whose evaluation value deviates from the best evaluation value for a hypothesis only by the constant. The time required for the recognition of a sentence can be vastly reduced by this "cutting off" for less promising hypotheses.

This "cutting off" for unfavourable hypotheses, i.e. on the grammatical level, can also be correspondingly applied directly to the comparison of the words, to be precise preferably additionally. For thiis, according to a further development of the invention, every entry in the second list whose first evaluation value is greater than the threshold value is delected. In this manner, the extent of the second list is normally continually reduced during the comparison with the successive incoming speech values, while on the other hand it is continually extended when tracing the more favourable hypotheses from the third list. In this manner, the extent of the second list remains restricted, so that the actual comparison procedure is executed quickly. A uniform evaluation for both cases is achieved by the use of the first evaluation value for the threshold value.

This threshold value can, however, only be determined when a new speech value has been compared with the reference values of all the words contained in the second list, so that a further pass is necessary for deleting entries in the second list in each case. This further pass can be dispensed with, according to a further development of the invention, in that the evaluation value of the preceding speech value is used for the determination of the threshold value as a smallest first evaluation value of the entries of the second list. However, since as a result of this a slightly smaller minimum first evaluation value results, the constant is increased slightly so that the same threshold value is essentially obtained here again as would result from the use of the last entries in the second list. The value of the constants is in any case dependent on the demands made of the speech system. If this constant is selected to be large, many hypotheses will be traced, so that the time required for the entire recognition of a sentence increases, whereas with a smaller value for the constant, the correct hypotheses may be lost in some very unfavourably spoken sentences, to that correct recognition can no longer occur.

The words or entries in the second list are deleted in particular when the actual word corresponding to the current speech values has little resemblance to a word contained in the second list. Since the last reference value of the last or a very similar word is, however, compared not only once but with several successive speech values, since for example the word ends can be spoken with a drawl, such a word would remain a relatively long time in the second list. Since a drawling of this kind is assumed to be only limited, however, it is favourable t delete such entries corresponding to completely recognized words as soon as possible. This can occur according to a further development of the invention in that a word is delected from the second list if the number of sequence numbers lying between the new sequence number and the sequence number stored at the word is greater than a limit value contained in the reference values for this word. In this manner, drawled ends of words can still be captured while, on the other hand, such words do not greatly impede to slow down the recognition procedure.

An entry into the second list is made from corresponding entries of the third list, namely where the second sequence begins with a word to be recognized. Particularly with an extensive vocabulary, a hypothesis can be continued with a multitude of various words all belonging to the same grammatical word class, for example, all verbs of the vocabulary. An entry is then also present in the third list for each individual word, so that the latter becomes very extensive. In order, in particular, to reduce the extent of the third list, it is expedient according to a further development of the invention that the first list contains assignments of predetermined syntactic categories to other syntactic categories and/or word classes instead of words and in that, for all entries of the new group where the second sequence begins with a word class, a reference to this word class instead of to reference data of a word is entered into the second list. All words to be entered into the second list or which must be compared with the next incoming speech values are then determined unambiguously and directly from the word class. The third list then contains only a single entry in each case for all words of this kind.

The extent of the second list can also be reduced in a similar manner when, namely, a reference is not entered separately to each of the words to be recognized, but likewise only to the word class, into the second list. In this case it is expedient that, with each reference in the second list, an auxiliary list is called up containing for each word class the references to the reference data of the words belonging to this class, and these references call up the corresponding reference values from the further list. The auxiliary list may have a very simple structure, since it only contains the assignment of a word class to the individual associated words.

For the realization of the recognition of speech signal, devices are known having a transducer for converting a spoken sentence into an electrical speech signal and for forming speech values, having a first memory containing specifications on syntactic categories of natural language and their assignment to further syntactic categories and/or specifications for words or word classes, having a further memory for reference values formed analogously to the speech values from sentences spoken earlier, and having a comparison device connected to an output of the transducer and to a data output of the further memory to supply comparison results from the comparison of speech values with reference values. The development of such an arrangement for carrying out the process according to the invention is characterized in that a second memory which stores the entries for the second list and a third memory which stores the entries for the third list are provided, the contents of the second memory specifying at least a part of the addresses of the further memory, and in that a controller is present which is set up such that it address the first, the second and the third memory and records data in the second and the third memory and reads out of these as well as out of the first memory and, on receiving a word end signal for at least one word, forms the new, the first and the second further entries for the third memory and subsequently the entries for the second memory and records in these and, after processing of the last speech signal, outputs the complete word string contained in the third memory with the smallest evaluation value to an output device. In this arrangement, it is particularly expedient that the controller is a processor, in particular a programmed microprocessor. Inasmuch as the third list contains references to word classes instead of the words themselves, it is moreover expedient that the output of the second memory is coupled to the address input of an auxiliary memory, the output of which is coupled to a partial address input of the further memory.

Exemplary embodiments of the invention will be explained below in more detail with reference to the drawing, in which

FIG. 1 shows a schematic overview of the two-step nature of the process according to the invention,

FIG. 2 shows a flow chart to clarify the process,

FIG. 3 shows the basic structure of the entries in the third list,

FIG. 4 shows the determining of the sequence of words or syntactic categories from an input sequence of speech values.

FIG. 5 shows a schematic block circuit diagram for carrying out the process according to the invention.

FIG. 1 shows a schematic diagram which clearly illustrates the two sections into which the process can be divided. The actual comparison of speech values, supplied via input 12, with reference values takes place in section 1. The comparison results for completely compared words are supplied via path 8 to section 2, where, on the one hard, these words are stored and, on the other hand, the comparison results of these in each case completely recognized words, together with the grammatical rules for this, are used to determined new words which can follow next on the basis of these grammatical rules, and these new words are supplied to section 1 via path 11, in order to determine or to complete the reference values with which the following speech values, supplied via input 12, are to be compared. In this manner, the process alternates continually between section 1 and 2, until the last speech value is compared. Subsequently, via output 14, the word sequence is output which has shown the best matching with the sequence of speech values and which moreover corresponds to the stored grammatical rules.

The speech values supplied via input 12 may, for example, be short-time spectra obtained from the speech signal at 10 ms intervals; they may however also be already further processed values, for example phonemes or diphones. The determining of the speech values, supplied via input 12, from the speech signal is not illustrated in more detail here, since this is performed in a usual manner and the details of this are not important for the invention.

The reference values with which the speech values are compared are obtained in the same manner as the supplied speech values from speech signals of previously spoken sentences, and they are stored in a memory or memory area 4. The actual comparison is symbolized by the circle 7, and it may be carried out, for example, in a manner known from German Offenlegungsschrift 3,215,868. In particular, varying speeds of speaking can thus be taken into account.

The reference values are selected in accordance with a list, which is designated hereinafter as the second list and which is stored in a memory or memory area 3. This second list contains references to words in coded form which, as described above, have been determined in section 2 of the process and specify which word can represent the speech values coming in via input 12 on the basis of the grammatical rules. Each word is incidentally assigned to a temporal sequence of reference values, the temporal successive processing of which is not illustrated in more detail or is contained in the comparison operation 7. The comparison result from the comparison operation 7 is again stored in the second list 3. As will be explained in more detail later, several words are "active" in each case, that is to say the associated reference values are compared with each incoming speech value, and the list in memory 3 contains yet further specifications in addition to the coded word. Since the various words are however mostly of differing lengths, that is to say contain a different number fo reference values, and since on the other hand various words are begun frequently at different times, the comparison of a word with the incoming speech values can be fully completed, that is to say the last reference value of the word has been compared with a speech value while the beginning or the middle area of other words is compared.

When a word has been complelely compared, therefore, it is supplied together with the further, still to be explained, specifications via path 8, which represents a step in the process, to section 2 and is used there to update a list, hereinafter designated as the third list, and is stored in a memory or memory area 5. This third list is modified yet again by further process steps, indicated by the arrows 9 and 10, arrow 10 specifying the consideration of a list which is hereinafter designated as the first list and is stored in a memory or memory area 6. This list has a structure such that the grammatical rules of speech are taken into account when determining the grammatically permissible next words to be compared. Arrow 9 indicates changes in the third list in the form of additional entries which are only formed on the basis of entries present in the third list.

The order of the overall process will be explained in more detail below with reference to the flow chart schematically represented in FIG. 2. The structure of the third list in memory 5 is of importance for this, which is indicated more fully in FIG. 3. This third list is subdivided into a number of groups of entries, in which entries 31 and 32 are specified for the first group with the sequence number 0, entries 41 and 42 are specified for the next group with the sequence number k and entries 51 to 54 are specified for the last group with the sequence number n. Each entry is subdivided into a number of sections a to g, which form quasi-columns over the row-by-row entries in the list. The sequence numbers are specified only once for each group for the sake of clarity. It is, however, actually contained in each entry in a corresponding section. This third list in FIG. 3 is first generated or filled up during the execution of the processing with the successive incoming speech values, that is to say initially the list is not present at all or is empty. The number of entries per group of the third list is not defined here, but is obtained from the course of the process on the basis of the incoming speech values.

In FIG. 2, symbol 20 represents the general start symbol. In block 21, the group 0 of the entries in the third list in FIG. 3 is generated before the first speech value arrives, to be precise on the basis of the first list containing the assignment of syntactic categories of various orders to each other and to words. A syntactic category is in particular a part of a sentence, for example the object, which may consist of a subordinate clause, of a noun with or without adjectives, of a pronoun or of an empty part of a sentence, that is to say may not be present at all. One line in the first list is available for the syntactic category "object" for each of these possibilities. The individual possibilities here in turn represent at least partly syntactic categories for which there are likewise various possibilities in the grammar, that is to say several lines in the first list, etc.

The first list with the syntactic categories and their assignment to each other and to words is defined for the process and determines which sentences can be recognized with which grammatical structures. This list is therefore already available at the beginning of the process and is not subsequently changed.

In every grammar, even in a very limited grammar only for simple sentences, a syntactic category is, however, always present, namely an initial category which comprises each recognizable sentence and for which the first list contains one or in general a plurality of assignments to other syntactic categories which represent virtually the most basic division of a sentence. These most basic syntactic categories in turn contain further assignments in the first list, etc., unit finally each assignment chain results in words to be recognized. An advantageous intermediate solution consists incidentally in that the strings of assignments are not allowed to end in the words to be recognized themselves, but in word classes, such as nouns, verbs, etc., for example. This reduces the extent of the first list, and the third list, and possibly also the second list, quite substantially. This will become even clearer during further explanation of the process. Only at transition 11 from the second process step to the first, that is to say when the next words to be recognized are transferred, is the word class in question broken down into the individual words. In doing so, each word of the word class in question can be entered into the second list, or only the word class is also entered into the second list, and from this word class the reference values of the associated individual words are called up, for example via an auxiliary memory.

In the first processing block 21 of the flow chart in FIG. 2, the entries for the group 0 are thus generated in the third list of FIG. 3. Any number may be selected as the sequence number of the individual groups; however, it is expedient to select this to begin in natural numeric order and continue upwards, so that here the sequence number 0 is selected.

The individual sections of each entry of the third list thus have the following meaning.

(a) This is a reference to a syntactic category of the first list, that is to say expediently the address of those lines of the first list in which a syntactic category occurs for the first time. Since all assignments to a syntactic category are frequently required one after the other, these are arranged successively in the first list, so that counting can be continued from the first line for the assignments of in each case one syntactic category.

(b) This section contains a specification for a sequence of already compared words and/or syntactic categories, wherein this sequence has moreover already been checked to see that it matches the grammatical rules according to the first list. This is automatically ensured in that the sequence specified in sectin b represent a part of the assignment to the syntactic category specified in section a.

(c) This section contains a specification for a sequence of words and/or syntactic categories which represents the rest of the assignment of the syntactic category in section a and hence specifies a sequence of possible words or categories still to be expected in the following speech signal.

(d) This section contains a further sequence number of a group, to be precise either of the current group or an earlier group, depending on how the entry in question of the third list is formed. This will become clearer from the further description of the process.

(e) This section contains accumulative evaluation value which specifies the total sum of all comparison sums between all previously arrived speech values and the total sequence of reference values belonging to the hypothesis of which the syntactic category specified in section a represents at least a part.

(f) This section contains an evaluation value which had been attained as the syntactic category was begun according to section a.

(g) This section contains a sequence of compared words corresponding to the sequence specified in section b. Since, in section b, however, also already only syntactic categories can be specified in part or completely, which therefore combine in each case a plurality of words or, to be more precise, a plurality of word types, the individual recognized words must be recorded separately in the correct sequence, which takes place on this section.

First of all, the assignments to the initial category in the first list are now entered in the entries 31, 32 etc. in the group 0. Here, therefore, section a contains the reference to this initial category, for example to the first line in the first list, section b contains an empty sequence, since no comparisons have been made yet, while section c contains the further syntactic categories assigned to the initial category. Section d contains the sequence number of the first group, i.e. 0. Sections e and f contain an initial value, expediently the value 0. Likewise, section g does not yet contain anything, or just an empty sequence.

When in this manner all assignments to the initial category to entries have been processed, the first syntactic category of the section c is checked for each entry, to see whether an assignment to further categories or words is present for this, and for each such assignment a further entry is made in group 0, where section a of the new entry contains the first category of section c of the old entry, section c contains the assigned sequence of syntactic categories or words or a mixture of these, while sections b and d to g contain the same values as for the first entry. Section c are likewise checked again in these extended entries to see whether it contains a sequence beginning with a syntactic category, etc., until finally only entries have been extended in which each sequence specified in section c begins with a word to be recognized, namely with one of the first possible words of all recognizable sentences. Since the words at the beginning of a sentence, however, all belong to one or a few different classes of word due to the grammar, just as the possible words at the further positions within the sentence, it is expedient only to specify the word classes in section b and c in the entries in the third list. Third reduces the extent of the third list quite substantially, as is readily apparent.

No further assignment is now present for this first word or word class, but this first word or word class in section c of the entries in question must be transferred into the second list, so that the first speech value can be compared with the reference values of the permissible first words of a sentence. Furthermore, the first evaluation value contained in section e, that is to say usually 0, as well as the associated sequence number of the group, that is to say in this case likewise the value 0, is transferred into this second list. This is the initial status of the second list, which however, is frequently changed during the course of the process.

After the second list has been created in this manner with the words to be compared and the third list has been created with the first group, the first speech value can be compared with reference values in block 22 in FIG. 2. The comparison result represents a measure of the matching or a distance value between the first speech value and the corresponding reference values, and this applies analogously also for the following speech values. Each new distance value determined is added in a manner known per se to the minimum comparison sum determined up until then. This occurs during the progressive course of the speech signal with a plurality of temporally sequential reference values of the same word in accordance with the principle of dynamic programming, as is described, for example, in the already mentioned German Offenlegungsschrift 3,215,868, in order in particular to be able to compensate for varying speeds of speaking, until finally the last reference value of a word has been compared, or more precisely, has been compared for the first time, since the next following speech value(s) is (are) likewise still compared with the last reference value. Incidentally, the sequence number is increased with each new speech value, but a new group with entries for a sequence number is only made when the associated speech value has been compared as mentioned with the last reference value of a word. Instead of this, an updating of the sequence number can only take place at each word end of this kind.

This takes place in block 23 in FIG. 2. In doing so, a new entry is made into the third list from such a completely compared word as well as from the sequence number stored at this word or the associated word class in the second list and from the evaluation value stored there, which is increased by the comparison sum of this word. For this, in the group, the sequence number of which is stored at the completely compared word or the associated word class in the second list, in the third list at least section c of each entry is read out and it is checked whether the sequence contained therein begins with the word or the associated word class which has just been completely compared. At least one such entry must be present in any case, since it is only from such an entry of the third list that the recongnized word or the associated word class in the second list can have come. For each such entry, for the group with the current sequence number in the third list, a new entry is generated which contains in section a the same reference to the syntactic category as the entry read out, in section b the sequence extended to include the completely compared word or the associated word class, in section c the

correspondingly abbreviated sequence, in section d the same sequence number as in section d of the entry read out and in section e an evaluation value equal to the sum of the first evaluation value from the entry read out and the comparison sum attained for the word in question, while the second evaluation value of the entry read out is transferred, and in section g the sequence contained therein is extended to include the completely compared word, specifically really only the compared word, not the word classes. At the first completely compared word after the beginning, some of these values of the new entry seem rather trivial; however, in the further course of the process, the derivation of the individual sections of each new entry has great significance.

Before a new entry of this kind is entered into the current group of the third list, it is however still checked whether the evaluation value determined for the section e is smaller than a threshold value. This threshold value is determined from the evaluation values of all entries currently contained in the second list, in that the smallest of these evaluation values is searched for and is increased by a fixed constant. As already mentioned earlier, for each word or for each word class in the second list, the smallest evaluation value in each case, that is to say the best possible evaluation for a sequence of words, is recorded which led to the word or the word class in question. In the continuation of the hypotheses which are currently active via a corresponding entry in the third list, although it is possible that a hypothesis which does not currently have the most favourable evaluation value turns out to be in the further continuation more favourable than other hypotheses, that is to say exhibits a lesser increase of the evaluation value, it is however assumed that this improvement is only limited. This limited improvement is taken into account by the constant when the threshold value is determined. If, therefore, a new entry has a first evaluation value, greater than this threshold value, it is assumed that this hypothesis can no longer lead to a word sequence with smallest overall evaluation even in the case of a favourable continuation. The value of the constant thus represents a compromise, since if this is selected to be too small, it may result in the most favourable word sequence being lost, because it is temporarily less favourable than others, while too large a constant leads to too many hypotheses being further traced, which increases the processing required for the recognition of the speech signal considerably. Moreover, in the latter case, the possibility of an incorrect recognition is increased. Overall, however, with a relatively high value of the constant, that is to say with a high threshold, a substantial reduction of the processing time is still achieved in contrast to the case when no threshold is taken into consideration.

Incidentally, the same threshold is also used in the comparison of the speech values with the reference values specified by the entries of the second list. In the majority of hypotheses, continuation can namely be made on the basis of the grammar with various words, which becomes particularly clear if word classes instead of words are entered in the second list. In most cases these possible words are substantially different, so that in the case of these words it is already evident after a comparison of only part of the reference data of this word that it does not match the word contained in the speech signal sufficiently. This is evident from the fact that for such words the sum from the associated evaluation value from the second list and the comparison results rises sharply. If such words are no longer taken into consideration as soon as this sum exceeds the threshold value, in that this word is then deleted in the second memory or, if word classes are stored in the second memory, the assignment of these badly matching words to the word class in question is no longer taken into consideration, then considerable processing time can likewise be saved.

A further reason for deleting a word in the second list or no longer taking the assignment of a word to a word class in the second memory into consideration arises from the fact that, as mentioned earlier, several successive speech values are compared with the last reference value of a word in order to take a drawled pronunciation at the end of a word into account. Although, if the word in question in the speech signal is not spoken with a drawl, the comparison sum when comparing the last reference value with further successive speech signals becomes rapidly larger, so that the sum of evaluation value and comparison sum exceeds the threshold value, however, the number of successive speech values compared with the last reference value can be reduced in that only an, if appropriate, word-dependent given number of successive comparisons are made and subsequently the word in question is deleted from the second memory. This can save further unnecessary processing time.

If the last reference value of several words has been compared simultaneously, that is to say with a certain speech value, this process step is performed separately for each word. When in this manner the corresponding new entries have been made in the third list for the completely compared word or all completely compared words, these individual new entries are checked to see whether section c begins with a syntactic category. This takes place in block 24 in the flow chart according to FIG. 2. If an entry is then found where section c fulfils this condition, to the group with the current sequence number are added first further entries which contain in section a the reference to the syntactic category of section c of the entry tested, in section b an empty sequence, in section c the sequence of further syntactic categories and/or words or word classes assigned to this category, in section d the current sequence number, in sections e and f the first evaluation value from section e of the entry tested and in section g an empty sequence. Before this entry, or each first entry, is actually made, however, it is checked whether an entry in the current group is not already present whose specifications in sections a to d completely match the entry to be made. If such an entry is indeed already present in the current group, it is checked whether the first evaluation value is greater than that of the entry to be made. If this is the case, the present entry is replaced by the entry to be made, and in the other case, the newly derived first further entry is not made. The comparison of the specifications in sections a to d ensures that in this manner two hypotheses are recombined which, although they have usually started from different points and have taken different paths, however now should find exactly the same continuation. In this case, only the hypothesis with the most favourable evaluation value up to now is therefore further traced, since another hypothesis cannot achieve a better evaluation value even at a later stage. A comparison of the first evaluation value of each of these first further entries with the threshold value is no longer necessary, since these first further entries

27

cannot have a higher evaluation value than the entries already present.

When this has been performed for all new entries in question of the group with the current sequence number, it is checked, in accordance with block 25 in the flow chart according to FIG. 2, whether in this group an entry is present in which section c contains an empty sequence, where therefore the sequence assigned to that category specified in section a has been completely compared.

In the case of such an entry, in the group whose sequence number is specified in this entry of the current group in section d, that earlier entry is searched for in which section c begins with that syntactic category specified in section a of the entry in question of the current group. A second further entry for the current group is then derived from the earlier entry, which entry contains in section a the syntactic category in section a of the earlier entry, in section b the sequence extended to include the syntactic category of the entry in question of the current group, in section c the correspondingly abbreviated sequence, in section d the sequence number of the earlier entry, in section e the sum of the first evaluation value of the earlier entry and the difference of the two evaluation values of the entry in question of the current group, in section f the corresponding evaluation value of the earlier entry and in section g the string of the word sequences from both entries. In this manner, the hypotheses for older, higher-ranking syntactic categories are continually verified. Before this second further entry is actually made into the third list, it can be checked analogously as described above whether the first evaluation value obtained by means of forming a sum has not become greater than the threshold value.

All entries of the current group are processed in this manner. In this process, only entries can arise in which section c contains an empty sequence, so that the preceding process step is carried out again for all entries of the current group, in which new categories are thus entered, and the verification of the older hypotheses is subsequently continued again etc., until no new syntactic category has been entered any more. This is indicated by the decision lozenge 26 in the flow chart in FIG. 2, which therefore performs the loop through blocks 24 and 25 until no new entry is made, when the process then moves to block 27. In block 27, all entries of the current group of the third list are run through again, but it is now checked whether an entry in section c is present whose associated sequence begins with a word or a word class. If such an entry is found, this word or word class is transferred into the second list together with the first evaluation value of this entry and the sequence number of the current group.

Before transferring into the second list, however, it is checked whether an entry with the same word or the same word class and the same sequence number is not already present therein. If such an entry is found, it is checked whether the entry present has a greater evaluation value than the entry to be made. If this is the case, the entry present is replaced by the entry to be made, otherwise no new entry is made into the second list. In this manner, the second list always only contains the entry of a particular word with the smallest evaluation value of in each case a sequence number, which is of significance for the abovementioned determining of the threshold value.

It is checked in lozenge 28 whether the last speech value has been input. If this is not the case, the process returns to block 22 and the next speech value is processed in the manner described. If, however, the last speech value has been reached, the process moves to block 29, in which the last group of the third list is searched for entries containing in section a the initial category, in section c an empty sequence and in section d the sequence number of the first group as a sign that a grammatically complete sentence has been compared, and if several such entries are present, that entry is used which has the smallest first evaluation value in section e, and from this entry the word sequence in section g is read out which represents the sentence recognized with the greatest probability. This completes the process.

FIG. 4 illustrates how new hypotheses are developed and old hypotheses are concluded with the assignments during the process. A sequence of syntactic categories, of which one is denoted by A, is assigned to the initial category, which is denoted here by S. This syntactic category A breaks down in turn into a sequence of a word w, a further syntactic category B and a further word z. The sequence of words x and y is in turn assigned to the syntactic category B. The first speech values a(1) and following are already assigned to the initial category via a corresponding tree. The speech values a(h) to a(k) have matched the word w best. The following speech values a(k+1) to a(i−1) were in turn the most similar to the word x. Analogously, the speech values a(i) and following were most similar to the word y, etc. . . After the comparison of both words x and y is completed, the process described verifies the syntactic category B, and hence the elements w and B forming the first sub-sequence are verified for the syntactic category A. As soon as the hypothesis for the word z is also completed, category A is completely verified, and a hypothesis is built up after the following category, not shown in more detail in FIG. 4. In this manner, in the process described, only hypotheses ever arise which are guaranteed to be grammatically correct from the beginning up to the current value, and this applies until the end of the speech signal, that is to say until the last speech value.

FIG. 5 illustrates schematically the block circuit diagram of a device for carrying out the process described. The acoustic speech signal is captured via the microphone 62 and converted into an electrical signal, and this is digitized in the device 64, and speech values are formed therefrom, which may be, for example, short-time spectra of the speech signal, LPC coefficients or even phonemes. These speech values are supplied one after the other as multibit data words via the connection 65 to the comparator 66.

The latter receives via the connection 67 reference data words from a memory 68 which contains for each word of a given vocabulary a sequence of reference data which has been formed analogously to the data words on the connection 65 in an earlier learning phase. The individual words in the reference data memory 68 are addressed either via the connection 85 from the memory 76 or via the connection *85b* from an auxiliary memory 80, which will both be explained later. The individual reference data within the word are addressed via the connection 63 from the comparator 66 or from a controller 72, depending on in which manner known per se the comparison is carried out in detail.

The comparison results formed by the comparator 66 are supplied to the controller 72 and processed there.

This controller 72, which may be a microprocessor, for example, addresses via the connection 75 a first memory 70 containing the specifications of the first list on the syntactic categories and their assignment to each other and to words or word classes, and supplies these via the connection 73 to the controller 72. This forms, in accordance with the process described, data for the third list, for which a third memory 78 is provided, which is addressed from the controller 72 via the connection 77 and which receives the data to be recorded via the connection 79. Analogously, the data read out from the third list are also supplied from the memory 78 via the connection 79 to the controller 72, so that the connection 79 is expediently of bidirectional nature.

The entries formed from the third list, in accordance with the process described, for the second list are supplied by the controller 72 to the memory 76, which is addressed via the connection 81, via the connection 83. When entering, the next free location in the second list, which may have become free due to the deletion of a word, is then addressed, and for the processing of a new speech value, supplied via the connection 65 to the comparator 66, all entries of the second list are addressed one after the other in memory 76. Provided that the second list contains in each entry a direct reference to in each case one word, this part of the entry is supplied via the connection 85 to the reference data memory 68 as an address. If the second list, however, contains in the entries references to word classes instead of individual words, which is more expedient in the case of an extensive vocabulary, this reference in question is supplied via the connection *85a* to the address input of an auxiliary memory 80 which contains at each address location the addresses of the reference data memory 68 for the individual words belonging to the word class, and these addresses are supplied one after the other via the connection *85b* to the reference data memory. Depending on the course of the comparison of the previous sequence of speech values with reference data, in each case the currently required reference data within the respective word are addressed via the connection 63.

As soon as the end of the sequence of reference data has been reached for a word, this is reported to the controller 72 via the connection 69. This then extends the third list in memory 78 and, if appropriate, forms new entries for further words or word classes to be compared in the second list in memory 76, as set out in the process described above and subsequently the next speech value on the connection 65 is again processed.

When the last speech value of the speech signal has been input, which may be determined, for example, by the recognition of a fairly long pause in speaking, the word sequence with the best evaluation value is read out from the third list in memory 68 and supplied via the connection 71 to an output device 74, which may be a display device or a printer or even a memory, for example.

What is claimed is:

1. A process for the recognition of speech signal derived from a continuous flow of spoken words, which speech signal comprises a temporal sequence of speech values, each of which values specifies a section of the speech signal; comprising:

comparing the speech values with predetermined stored reference values, a group of which reference values represents one word of a predetermined vocabulary for forming an initial evaluation value;

summing the comparison results over various sequences of combinations of reference values and speech values per sequence whose order is permissible in accordance with a predetermined stored first list containing, for predetermined syntactic categories, at least one assignment per category to a combination of further syntactic categories and/or words for forming a cumulative evaluation value;

generating a second list and a third list the second list including references to the reference values of all those words which are compared with the respective next speech value as well as a sequence number per word, and the third list including, for each speech value which has been compared with the last reference value of at least one word, a plurality of entries, each entry including a current sequence number and:

(a) a reference to a syntactic category of the first list,

(b) a first specification for a sequence of compared words and/or syntactic categories which are assigned to sequences of already compared speech values,

(c) a second specification for a sequence of words and/or syntactic categories which can be assigned to subsequent speech values on the basis of the first list,

(d) a further sequence number assigned to the respective entry,

(e) a first cumulative evaluation value,

(f) a second initial evaluation value and

(g) a sequence of compared words;

determining a new sequence number at least after every comparison of a new speech value with the last reference value of at least one word, and after each such comparison, searching through the group of entries of the third list associated with the sequence number stored in the second list at this word for such entries in which the sequence contained in the second specification begins with the compared word, and deriving a new entry for each such entry present, for the new group of the third list associated with the new sequence number;

making a first further entry in the new group for each new entry in which the abbreviated sequence contained in the second specification begins with a syntactic category, for which at least one assignment is present in the first list, and, deriving a second further entry for the new group for each of the new and first further entries of the new group for which the second specification contains an empty sequence;

repeating the steps of deriving and making the first and second further entries alternately until, after at least one first further entry, no second further entry occurs;

entering a reference to the reference data of the first word of each entry of the new group in which the second sequence begins with a word to be recognized;

comparing the next speech value with the reference values of all words contained in the second list;

repeating the process steps until the last speech value of the speech signal to be recognized has been processed; checking the last group of the third list for all entries containing: a reference to the syntactic initial category, an empty sequence, and a sequence number; and

reading out the sequence of compared words from those entries having the smallest first evaluation value.

2. A process according to claim 1, wherein:

before the comparison of the first speech value, the first group contains:

a plurality of first entries each entry including: a reference to a syntactic initial category, an empty sequence as a first specification, another of the combinations assigned to the initial category as a second specification, and an initial value for the two evaluation values; and

further entries including all the catagories which can be derived from each of the combinations of the second specification with a corresponding combination; and

wherein each new entry in the first specification contains a sequence extended to include the compared word, a sequence abbreviated to exclude the compared word in the second specification, the evaluation value incremented by the sum of the comparison results of the word as a first evaluation value, the sequence extended to include the compared word as a sequence of compared words, and the values of the entry present;

wherein each of the first further entries includes a reference to the syntactic category of the new entry, from which this first further entry is derived, an empty sequence in the first specification, another combination assigned to the syntactic category as a sequence in the second specification, the new sequence number as a further sequence number, the evaluation value of the new entry for both evaluation values, and an empty sequence as a sequence of compared words;

reading out the earlier entry for each second further entry, from that group specified in the further sequence number of the relevant new or further first entry, where the sequence associated with the second specification begins with the new syntactic category to which this new or first further entry of the new group contains a reference, wherein the second entry contains: the reference to the syntactic category of the earlier entry, a sequence extended to include the syntactic category of the current new or first further entry in the first specification, a sequence abbreviated to exclude the same syntactic category in the second specification, the sequence number of the earlier entry as a further sequence number, the sum of the first evaluation value of the earlier entry and the difference between the two evaluation values of the current entry as a first evaluation value, the corresponding evaluation value of the earlier entry and the sequence of compared words of the earlier entry extended to include the sequence of compared words of the entry of the current group as a second evaluation value; and

entering into the second list each reference to the reference data, the associated first evaluation value, and the sequence number of the relevant entry.

3. A process according to claim 2 comprising making each first and second further entry only if no entry is present in the new group, which entry contains the same reference, the same first and second specification and the same further sequence number and in which the first evaluation value is smaller than the first evaluation

value of the intended further entry; and if such an entry is already present but with a greater evaluation value, deleting such entry.

4. A process according to claim 2 comprising entering a reference to reference data into the second list only if no reference to the reference data of the same word and the same sequence number with a smaller evaluation value is already present in said list; and if such entry is already present but with a greater evaluation value, deleting such entry.

5. A process according to claim 2, comprising making each new and each first and second further entry only if its first evaluation value is smaller than a threshold value which is equal to the smallest first evaluation value, extended by a constant, of all entries currently contained in the second list.

6. A process according to claim 5, comprising deleting every entry in the second list whose first evaluation value is greater than the threshold value.

7. A process according to claim 6, comprising determining from the evaluation value of the preceding speech value, the threshold value as a smallest first evaluation value of the entries of the second list.

8. A process according to claim 2, comprising deleting a word from the second list if the number of sequence numbers lying between the new sequence number and the sequence number stored at the word is greater than a limit value contained in the reference values for this word.

9. A process according to claim 1, comprising assigning predetermined syntactic categories to other syntactic categories and/or word classes in said first list; and entering a reference to a word class in said second list for all entries of the new group where the second sequence begins with a word class.

10. A process according to claim 9, comprising calling up an auxiliary list with each reference in the second list, an auxiliary list is called up containing for each word class the reference to the reference data of the words belonging to this class, and these references call

up the corresponding reference values from the further list.

11. Apparatus for carrying out the process according to claim 1, comprising input means for receiving a spoken sentence in the form of an electrical speech signal;

conversion means connected to said input means for forming speech values; a first memory containing specifications on syntactic categories of natural language and their assignment to further syntactic categories and/or specifications for words or word classes;

a further memory for reference values formed analogously to the speech values from sentences spoken earlier;

comparison means connected to an output of the conversion means and to a data output of the further memory for supplying comparison results from the comparison of speech values with reference values;

a second memory for storing the entries for the second list specifying at least a part of the address of the further memory; and

a third memory for storing the entries for the third list;

controller means for addressing the first, the second and the third memory and for recording data in the second and the third memory and reading out of the first, second and third memory and on receiving a word end signal for at least one word, forming the new first and second further entries for the third memory and subsequently the entries for the second memory and recording in these; and, output means for outputting after processing the last speech signal the complete word string contained in the third memory with the smallest evaluation thereof.

12. Apparatus according to claim 11, wherein the controller comprises a programmed microprocessor.

13. Apparatus according to claim 11, comprising an auxiliary memory having an address input coupled to an output of the second memory and an output coupled to a partial address input of the further memory.

* * * * *

Synonyms

US005371807A

United States Patent [19]

Register et al.

[11] Patent Number: 5,371,807

[45] Date of Patent: Dec. 6, 1994

R

(2)

[54] METHOD AND APPARATUS FOR TEXT CLASSIFICATION

[75] Inventors: Michael S. Register; Narasimhan Kannan, both of Colorado Springs, Colo.

[73] Assignee: Digital Equipment Corporation, Maynard, Mass.

[21] Appl. No.: 855,378

[22] Filed: Mar. 20, 1992

[51] Int. Cl.5 .... G06K 9/72; G06F 15/38

[52] U.S. Cl. 382/14; 382/40; 364/419.07; 364/419.08

[58] Field of Search 382/36-38, 382/40, 14, 15; 395/21-23; 364/419

[56] References Cited

U.S. PATENT DOCUMENTS

4,674,065	6/1987	Lange et al.	382/40
4,682,365	7/1987	Orita et al.	382/14
4,754,489	6/1988	Bokser	382/14
4,876,731	10/1989	Loris et al.	382/40
5,050,218	9/1991	Ikeda et al.	382/1
5,056,021	10/1991	Ausborn	364/419
5,083,268	1/1992	Hemphill et al.	364/419
5,128,865	7/1992	Sadler	364/419
5,146,406	9/1992	Jensen	364/419

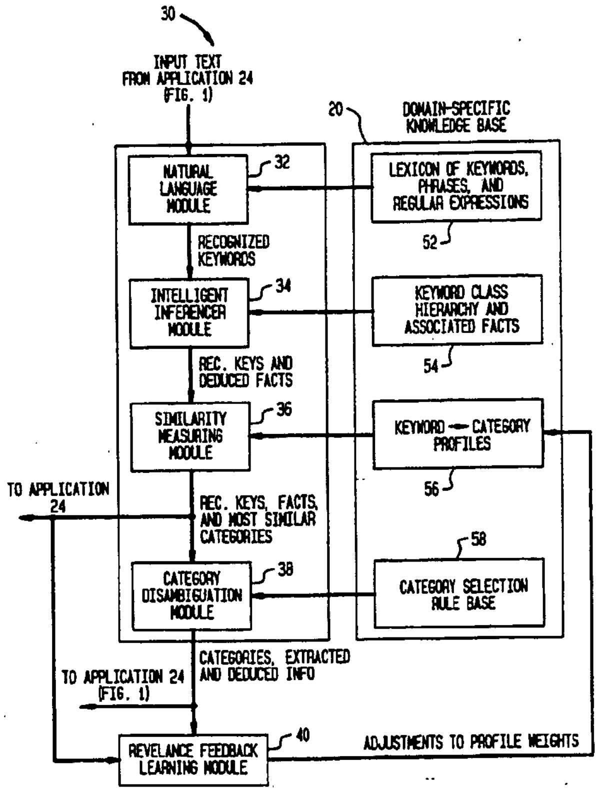
Primary Examiner—Leo H. Boudreau
Assistant Examiner—Chris Kelley

Attorney, Agent, or Firm—Kenyon & Kenyon

[57] ABSTRACT

A text classification system and method that can be used by an application for classifying natural language text input into a computer system having a domain specific knowledge base that includes a knowledge base having a plurality of categories. The text classification system classifies input natural language input text by first parsing the natural language input text into a first list of recognized keywords. This list is then used to deduce further facts from the natural language input text which are then compiled into a second list. Next, a numeric similarity score for each one of the plurality of categories in the knowledge base is calculated which indicates how similar one of the plurality of categories is to the natural language input text. A dynamic threshold is then applied to determine which ones of the plurality of categories are most similar to the recognized keywords of the natural language input text. A third list is compiled of the ones of the plurality of categories determined to be most similar to the recognized keywords. An optional rule base can be utilized to further refine the determination of which ones of the plurality of categories are most similar to the recognized keywords of the natural language input text. Also, an optional learning capability can be added to improve the accuracy of the text classification system.

24 Claims, 6 Drawing Sheets



30
INPUT TEXT FROM APPLICATION 24 (FIG. 1)
20
DOMAIN-SPECIFIC KNOWLEDGE BASE
NATURAL LANGUAGE MODULE
32
LEXICON OF KEYWORDS, PHRASES, AND REGULAR EXPRESSIONS
52
RECOGNIZED KEYWORDS
INTELLIGENT INFERENCER MODULE
34
KEYWORD CLASS HIERARCHY AND ASSOCIATED FACTS
54
REC. KEYS AND DEDUCED FACTS
SIMILARITY MEASURING MODULE
36
KEYWORD — CATEGORY PROFILES
56
TO APPLICATION 24
REC. KEYS, FACTS, AND MOST SIMILAR CATEGORIES
58
CATEGORY DISAMBIGUATION MODULE
38
CATEGORY SELECTION RULE BASE
TO APPLICATION 24 (FIG. 1)
CATEGORIES, EXTRACTED AND DEDUCED INFO
REVELANCE FEEDBACK LEARNING MODULE
40
ADJUSTMENTS TO PROFILE WEIGHTS

FIG. 1

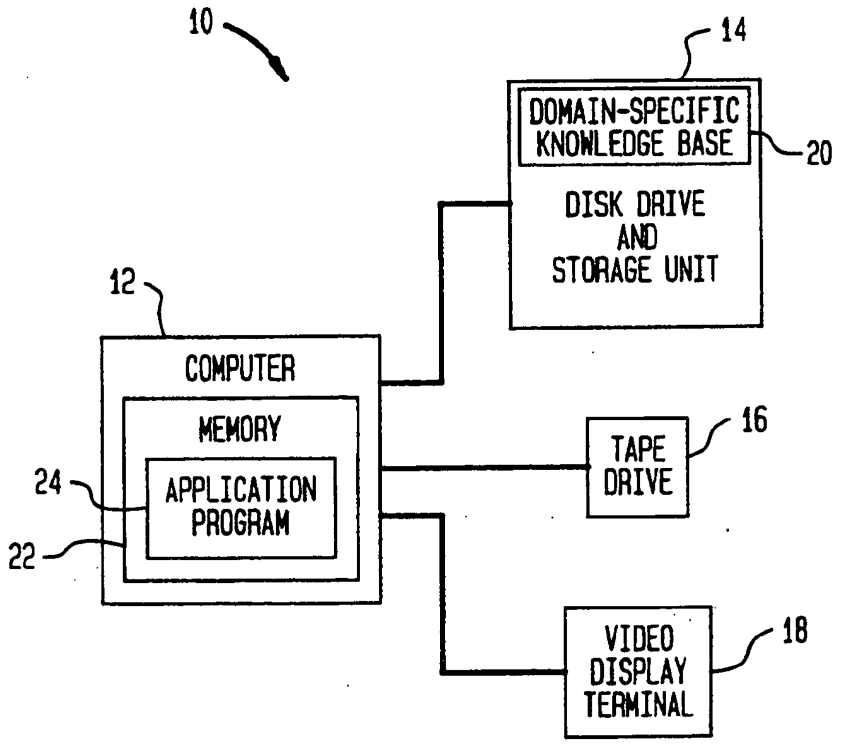

10
14
DOMAIN-SPECIFIC KNOWLEDGE BASE
20
DISK DRIVE AND STORAGE UNIT
12
COMPUTER
MEMORY
APPLICATION PROGRAM
24
22
TAPE DRIVE
16
VIDEO DISPLAY TERMINAL
18

FIG. 2

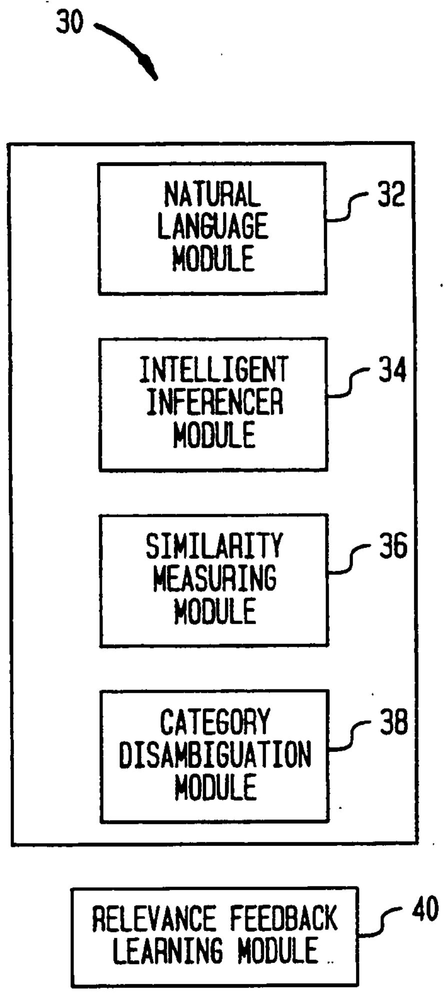

30
NATURAL LANGUAGE MODULE
32
INTELLIGENT INFERENCER MODULE
34
SIMILARITY MEASURING MODULE
36
CATEGORY DISAMBIGUATION MODULE
38
RELEVANCE FEEDBACK LEARNING MODULE
40

FIG. 3

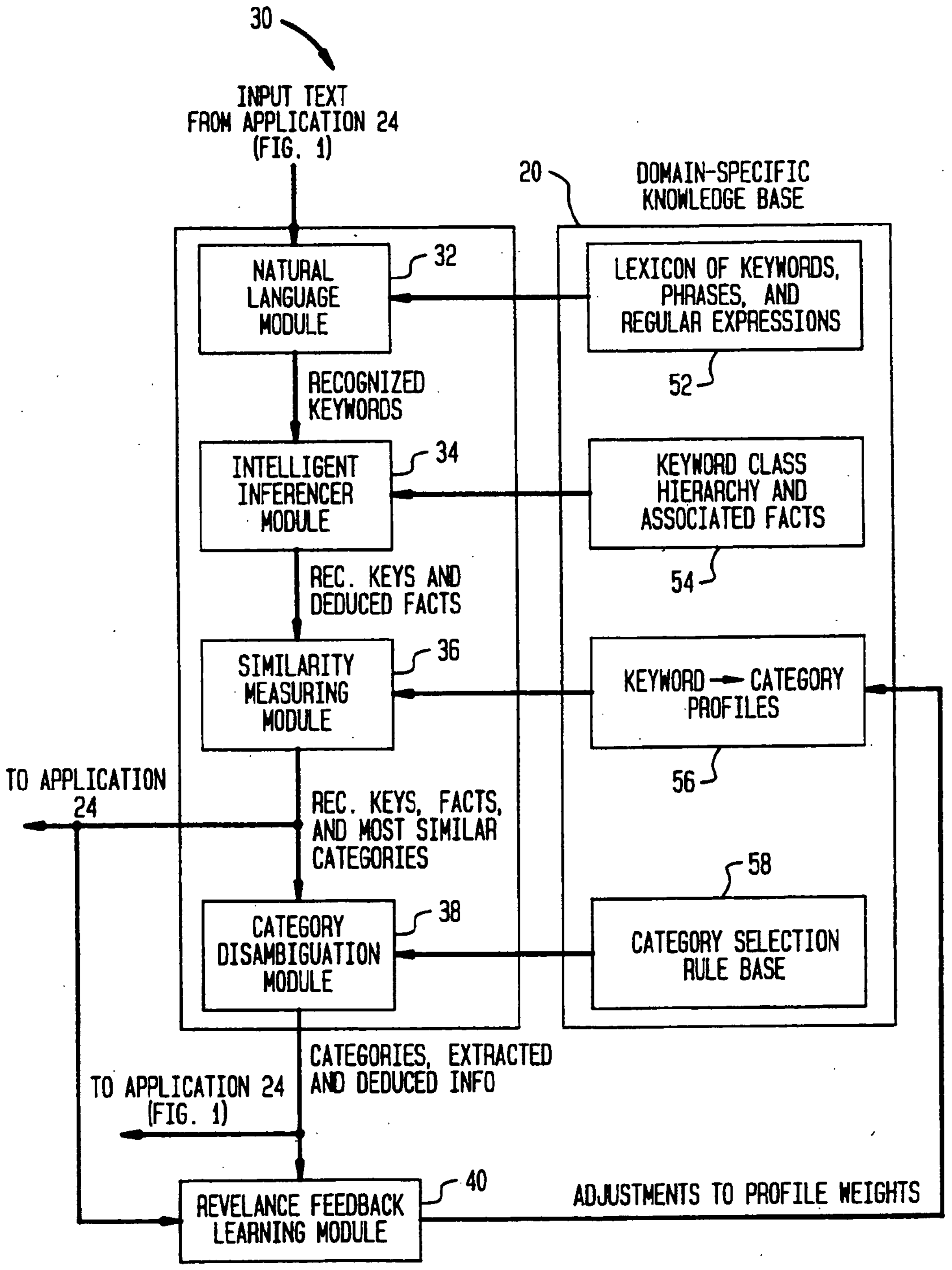

30
INPUT TEXT FROM APPLICATION 24 (FIG. 1)
20
DOMAIN-SPECIFIC KNOWLEDGE BASE
NATURAL LANGUAGE MODULE
32
LEXICON OF KEYWORDS, PHRASES, AND REGULAR EXPRESSIONS
52
RECOGNIZED KEYWORDS
INTELLIGENT INFERENCER MODULE
34
KEYWORD CLASS HIERARCHY AND ASSOCIATED FACTS
54
REC. KEYS AND DEDUCED FACTS
SIMILARITY MEASURING MODULE
36
KEYWORD → CATEGORY PROFILES
56
TO APPLICATION 24
REC. KEYS, FACTS, AND MOST SIMILAR CATEGORIES
58
CATEGORY DISAMBIGUATION MODULE
38
CATEGORY SELECTION RULE BASE
CATEGORIES, EXTRACTED AND DEDUCED INFO
TO APPLICATION 24 (FIG. 1)
REVELANCE FEEDBACK LEARNING MODULE
40
ADJUSTMENTS TO PROFILE WEIGHTS

FIG. 4

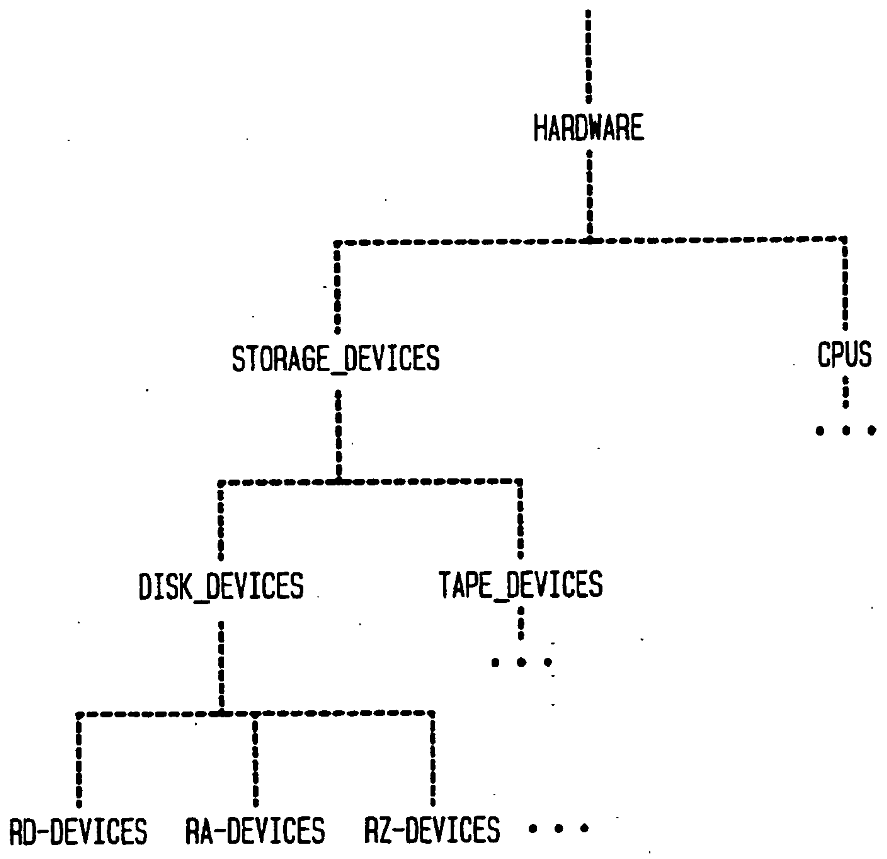

HARDWARE
STORAGE_DEVICES
CPUS
• • •
DISK_DEVICES
TAPE_DEVICES
• • •
RD-DEVICES
RA-DEVICES
RZ-DEVICES
• • •

FIG. 5

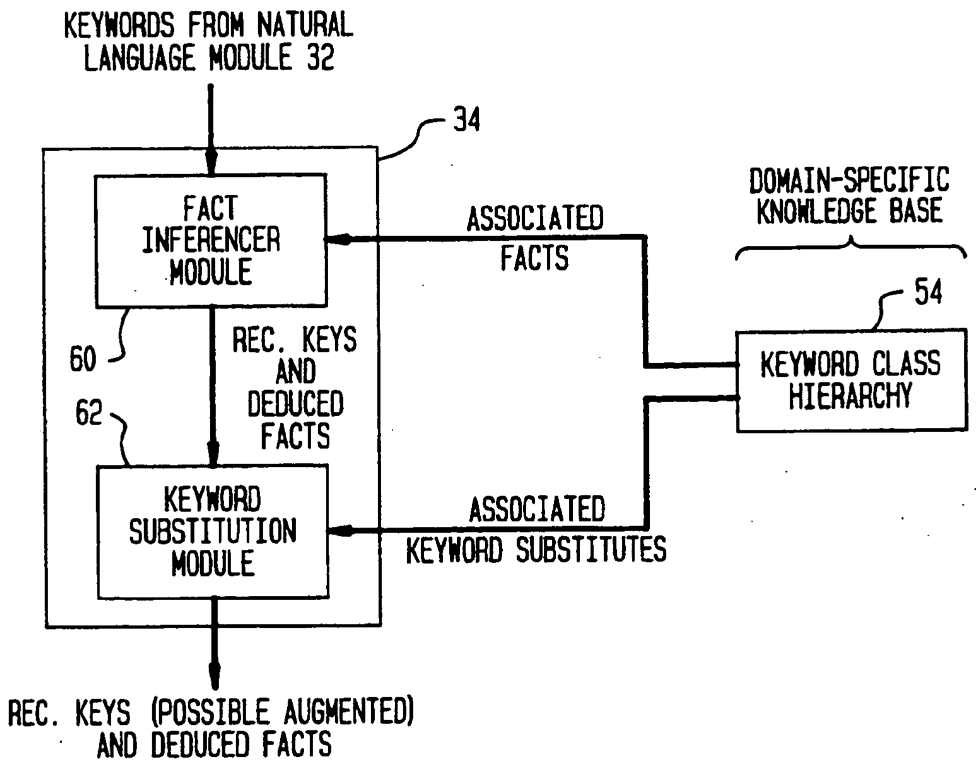
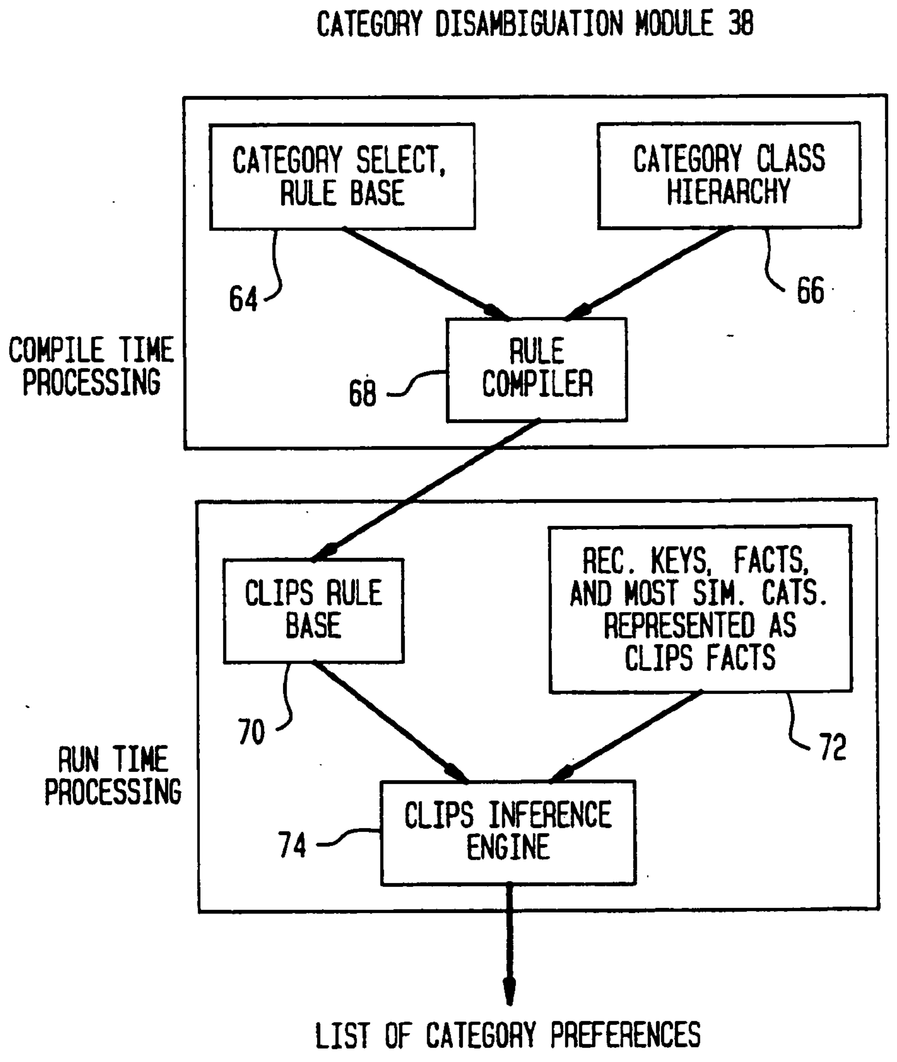

KEYWORDS FROM NATURAL LANGUAGE MODULE 32
34
FACT INFERENCER MODULE
60
REC. KEYS AND DEDUCED FACTS
62
KEYWORD SUBSTITUTION MODULE
ASSOCIATED FACTS
DOMAIN-SPECIFIC KNOWLEDGE BASE
54
KEYWORD CLASS HIERARCHY
ASSOCIATED KEYWORD SUBSTITUTES
REC. KEYS (POSSIBLE AUGMENTED) AND DEDUCED FACTS

FIG. 6


CATEGORY DISAMBIGUATION MODULE 38
CATEGORY SELECT, RULE BASE
CATEGORY CLASS HIERARCHY
64
66
COMPILE TIME PROCESSING
68
RULE COMPILER
CLIPS RULE BASE
REC. KEYS, FACTS, AND MOST SIM. CATS. REPRESENTED AS CLIPS FACTS
70
72
RUN TIME PROCESSING
74
CLIPS INFERENCE ENGINE
LIST OF CATEGORY PREFERENCES

METHOD AND APPARATUS FOR TEXT CLASSIFICATION

FIELD OF THE INVENTION

The present invention is directed to text classification and, more particularly, to a computer based system for text classification that provides a resource that can be utilized by external applications for text classification.

BACKGROUND OF THE INVENTION

The growing volume of publicly available, machine-readable textual information makes it increasingly necessary for businesses to automate the handling of such information to stay competitive. By automating the handling of text, businesses can decrease costs and increase quality in performing tasks that require access to textual information.

A commercially important class of text processing applications is text classification systems. Automated text classification systems identify the subject matter of a piece of text as belonging to one or more categories from a potentially large predefined set of categories. Text classification includes a class of applications that can solve a variety of problems in the indexing and routing of text.

Routing of text is useful in large organizations where there is a large volume of individual pieces of text that needs to be sent to specific persons (e.g., technical support specialists inside a large customer support center). Indexing text is useful in attaching topic labels to information and partitioning the information space to aid information retrieval. Indexing can facilitate the retrieval of information based upon the contents of text rather than boolean keyword searches from databases that include information such as news articles, federal regulations, etc.

A number of different approaches have been developed for automatic text processing. One approach is based upon information retrieval techniques utilizing boolean keyword searches. This approach, however, has problems with accuracy. A second approach borrows natural language processing from artificial intelligence technology to achieve higher accuracy. While natural language processing improves accuracy based upon an analysis of the meaning of input text, speed of execution and range of coverage becomes problematic when such techniques are applied to large volumes of text.

Others have recognized the foregoing shortcomings and have attempted to reach a middle ground between information retrieval techniques and natural language/-knowledge-based techniques to achieve acceptable accuracy without sacrificing speed of execution or range of coverage. This has been accomplished through predominantly rule based systems which parse the input text using natural language morphology techniques, attempt to recognize concepts in the text, and then use a rule base to map from identified concepts to categories.

Text classification systems which rely upon rule-base techniques also suffer from a number of drawbacks. The most significant drawback being that such systems require a significant amount of knowledge engineering to develop a working system appropriate for a desired text classification application. It becomes more difficult to develop an application using rule-based systems because all the requisite knowledge is placed into a rule base. By

doing this, a knowledge engineer must spend a significant amount of time tuning and experimenting with the rules to arrive at the correct set of rules to ensure that the rules work together properly for the desired application.

Another shortcoming in the foregoing systems is that there is no built in mechanism to allow the knowledge base portion of a text classification system to learn from the input text over time to thereby increase system accuracy. The addition of a learning component to enhance the accuracy of a text classification system would be desirable to improve the performance of the system over time.

SUMMARY OF THE INVENTION

The present invention provides a method and system for performing text classification. Specifically, the system provides a core structure that performs text classification for external applications. It provides a core run time engine for executing text classification applications around which the knowledge needed to perform text classification can be built.

Generally, the operating environment of the present invention includes a general purpose computer system which comprises a central processing unit having memory, and associated peripheral equipment such as disk drives, tape drives and video display terminals. The system of the present invention resides in either memory or one of the storage devices. It is invoked by an application running on the central processing unit to classify input text. A knowledge base is maintained on the disk drive or some other storage medium in the computer system.

The method of classifying text according to the present invention begins upon acceptance of natural language input text which can be supplied by an external application. The input text is then parsed into a first list of recognized keywords which may include, e.g., words, phrases and regular expressions. The first list is used to deduce further facts from the natural language input text which are useful in classifying the input text. The deduced facts are then compiled into a second list. Then, utilizing the first list, the present invention calculates a numeric similarity score for each one of a plurality of categories in the knowledge base which indicates how similar one of the plurality of categories is to the recognized keywords in the first list. A dynamic threshold is then applied to determine which ones of the categories are most similar to the recognized keywords of the natural language input text. The result is a third list which includes the categories that the recognized keywords are most similar. At this point, the text classification operation of the present invention is complete and the first, second and third lists can be passed on to the external application for application specific processing.

The architecture of the text classification system of the present invention comprises a natural language module, an intelligent inferencer module and a similarity measuring module. The natural language module extracts as much information as possible directly from natural language input text received by the text classification system from an external application. The intelligent inferencer module deduces any and all relevant information that is implicitly contained in the natural language input text. The similarity measuring module calculates a numeric similarity score for each one of the plurality of categories and applies a dynamic threshold to the plurality of categ ries to ascertain which categories are potentially most similar to the natural language input text.

A domain specific knowledge base comprising a lexicon of keywords, a class hierarchy organization for keywords and a class hierarchy organization for categories is utilized by the text classification system of the present invention. The knowledge base is provided by the external application that is utilizing the text classification system of the present invention. By allowing keyword and category classes, the present invention simplifies the maintenance of the lexicon and an optional rule base. Accuracy is also improved by allowing multiple facts to be inferred from single keyword classes.

An optional category disambiguation module can be added to the system of the present invention to further refine the results obtained by the similarity measuring module. Under such circumstances, the domain specific knowledge base can be adapted to include the optional rule base. By making the category disambiguation module and the rule base optional, the present invention provides a text classification application developer more flexibility by allowing the developer to decide whether or not to include the rule base. While eliminating the category disambiguation module and the rule base may result in some loss of accuracy, the trade-off would be that development of an application is greatly simplified.

If, however, an application developer decides to utilize the category disambiguation module and the rulebase, the task is simple and straightforward because most of the processing and comparison of the input text is performed upstream in the architecture thereby greatly reducing the importance of the rule-base in the text classification process.

The system of the present invention can also be adapted to include an optional relevance feedback learning module as an add-on to the system of the present invention to learn over time to increase system accuracy. It can operate independently of the text classification system, e.g., in a batch mode. The relevance feedback learning module utilizes information passed to it by the system to adjust values stored in a category profile knowledge base. Such information may include a category determined most relevant to a given natural language input text, a category determined most relevant to the same natural language input text by an external source, e.g., a human expert, (the categories may or may not be the same), and a list of keywords that provide evidence for the categories selected along with the amount of evidence they provide.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 illustrates an exemplary computer system for implementing a text classification system according to the present invention.

FIG. 2 illustrates an exemplary architecture of the modules utilized in a text classification system according to the present invention.

FIG. 3 shows the exemplary architecture illustrated in FIG. 2 together with an exemplary domain specific knowledge base.

FIG. 4 illustrates an exemplary portion of the keyword class hierarchy.

FIG. 5 illustrates an exemplary embodiment of the modules that comprise the intelligent inferencer module illustrated in FIG. 2.

FIG. 6 illustrates an exemplary implementation of the category disambiguation module illustrated in FIG. 2.

DETAILED DESCRIPTION

Referring now to the drawings, and initially to FIG. 1, there is illustrated an exemplary embodiment of a system for implementing the present invention. The system 10 comprises a computer 12 having a memory 22 associated therewith and with associated peripheral equipment such as a disk drive and storage unit 14, a tape drive 16 and a video display terminal 18. The computer 12 is generally any high performance computer such as a Digital Equipment Corporation VAX 6000-100. In conjunction with the computer 12, a domain specific knowledge base 20 that includes application-specific information is stored on the disk drive 14 and an application program 24 is stored in the memory 22.

Referring now to FIG. 2, there is illustrated an exemplary architecture for a text classification system 30 of the present invention. The system 30 comprises a natural language module 32, an intelligent inferencer module 34 and a similarity measuring module 36. An optional category disambiguation module 38 and an optional relevance feedback learning module 40 are also shown in FIG. 2. Modules 32, 34 and 36 (and 38, if selected to be part of the system) comprise what is hereinafter referred to as "the run time system" of the present invention. These modules are referred to as the run time system because collectively, they are invoked by the computer 12 (FIG. 1) to process and classify natural language text received from an external source, e.g., the application 24.

FIG. 3 illustrates the system 30 of FIG. 2 with the domain specific knowledge base 20 of FIG. 1. As illustrated in FIG. 1, the knowledge base 20 is shown as being stored on the disk drive 14. It should be understood that it could also be stored in the memory 22 (FIG. 1) or any other appropriate storage device coupled to the computer 12. The knowledge base 20 is external to the system 30. The information stored in the knowledge base 20 is provided by an applications programmer who is charged with developing the application 24 that is utilizing the system 30 to perform text classification functions. The modules which comprise the domain specific knowledge base 20 are a lexicon 52, a keyword class hierarchy 54, keyword/category profiles 56 and an optional category selection rule base 58 (utilized when the optional category disambiguation module 38 is used).

Each of the modules of the system 30 and the components of the knowledge base 20 are briefly discussed below.

The main function of the natural language module 32 is to extract as much information as possible directly from natural language input text. The input text can be any machine-readable natural language text as determined by the external application 24. The natural language module 32 uses the lexicon 52, which comprises keywords which can include, for example, words, phrases, and regular expressions, to identify all recognized keywords in the natural language input text. Specifically, the module 32 extracts all the relevant information that is explicitly contained in the input text. The natural language module 32 passes a list of all the recognized keywords to the intelligent inferencer module 34.

An example of a natural language module of the type described above is disclosed in U.S. patent application

Ser. No. 07/729,445, entitled "Method and Apparatus for Efficient Morphological Text Analysis using a High Level Language for Compact Specification of Inflectional Paradigms," (hereinafter "the Morphological Text Analysis patent application") filed Jul. 12, 1991 and assigned to Digital Equipment Corporation. This application is expressly incorporated herein by reference.

The list of recognized keywords passed to the intelligent inferencer module 34 is used to deduce any and all relevant information that is implicitly contained in the input text. To accomplish this task, the intelligent inferencer module 34 uses the keyword class hierarchy 54 to deduce further facts from the information explicitly stated in the input text. Keywords are grouped into classes in the keyword class hierarchy 54. Each class has associated facts that are true when a member of the class is identified in the input text.

For example, the input text may mention problems with a specific type of disk device but not explicitly mention that the problems are with a disk. The keyword class hierarchy 54 can include a class called "DISK DEVICES" with specific disks as members. The fact "(DEVICE TYPE=DISK)" can be attached to this class. When a specific disk device is identified, the fact "(DEVICE TYPE=DISK)" can be inferred even though the word "disk" was not explicitly mentioned in the input text. The intelligent inferencer module 34 also performs word substitutions in key phrases. The intelligent inferencer module 34 passes the list of recognized keywords and a list of all the extra facts that could be deduced from the recognized keywords to the similarity measuring module 36.

The list of recognized keywords extracted from the input text passed to the similarity measuring module 36 is used to calculate a numeric similarity score for each predefined category. Each score indicates how similar a given category is to the input text. The similarity measuring module 36 uses a knowledge base of keyword/category profiles 56 to determine the similarity score. Each category in the knowledge base of keyword/category profiles 56 has an associated profile. The profile tells the similarity measuring module 36 which keywords provide evidence for the given category. Associated with each keyword in a profile is a numeric weight called a "profile weight" that tells the similarity measuring module 36 the amount of evidence a keyword provides for the given category. The module 36 determines profile weights and combines the profile weights to arrive at similarity scores for all the categories. Once the similarity scores have been calculated, a dynamic threshold is applied to all of the categories defined in the domain specific knowledge base 20. Those categories whose similarity scores are below the threshold are discarded from consideration as being potentially most similar to the input text. The categories whose similarity scores are above the threshold are compiled into a list and are passed to the next module or directly to the external application 24 (not shown), along with the list of extracted keywords and the list of deduced facts, if there are any.

The list of most similar categories, the list of extracted keywords, and the list of deduced facts, if any, can then either be passed directly out to the external application 24, to the optional category disambiguation module 38 or to the optional relevance feedback learning module 40. If a rule base is desired for a particular application, the information is passed to the category disambiguation module 38. The module 38 uses the category selection rule base 58 to select certain categories over other categories based on the list of recognized keywords and the list of deduced facts. This module 38 further refines the list of the most similar categories and passes it, along with the list of recognized keywords and the list of deduced facts to the external application 24 and, if desirable, the optional relevance feedback learning module 40.

The relevance feedback learning module 40 is an add-on to the run time system of the present invention. It can operate independently of the run time system, e.g., in a batch mode. The input to the relevance feedback learning module 40 comprises the category determined most relevant to a given input text, the category determined most relevant to the same input text by an external source, e.g., a human expert, (the categories may or may not be the same), and the list of recognized keywords that provide evidence for the categories selected along with the amount of evidence they provide. The module 40 then takes this information and adjusts the profile weights in the keyword/category profiles 56 accordingly.

The task that the natural language module 32 of the run time system of the present invention performs is to extract all the relevant information that is explicitly contained in a natural language input text. To accomplish this task, the module 32 uses the lexicon 52. The lexicon 52 contains all the information that is considered relevant for extraction purposes.

A brief description of the processing performed by the natural language module 32 is set forth below. For a complete description, the reader is referred to the Morphological Text Analysis patent application, referred to above, which is expressly incorporated herein by reference.

The natural language module 32 allows the inclusion of single word nouns and multiple word noun phrases into the lexicon 52. The natural language module 32 will recognize the root form of a noun or noun phrase as well as morphological variants of the root, e.g., plural form of the root noun or noun phrase. It also allows synonyms of a keyword to be entered into the lexicon 52 which are useful when defining keyword classes or when writing disambiguation rules.

Single word verbs can also be included in the lexicon 52. The root form of a verb must be entered into the lexicon 52. This way, the module 32 will not only recognize the root form, but morphological variants as well. For example, the verb "crash" in the lexicon 52 will identify "crashes", "crashing", and "crashed".

A limited form of multiple word verb phrases are allowed into the lexicon 52. In this case, a verb phrase is considered to be a single word verb combined with a single word noun or noun phrase subject/object (e.g., "Analyze Disk").

When keyword matching is performed for a verb phrase, each sentence in the input text is reviewed separately. For each sentence, the natural language module 32 tries to find the verb contained in the verb phrase. If the verb is found, it then looks to see if the noun or noun phrase contained in the verb phrase is present in the sentence. If both the verb and the noun phrase are found in the same sentence, then the entire verb phrase has been identified. For example, if the lexicon 52 contains the verb phrase "Analyze Disk." One of the sentences in the input text that the present invention is parsing is the following: "I need help analyzing this damaged disk." The natural language module 32 will first identify the single keywords "analyze" and "disk" (analyzing is a morphological variant of analyze). Then it will notice that "analyze" is the verb part of a verb phrase. It will then search the list of recognized keywords for that sentence for the noun part of the phrase (in this case the word "disk"). Since "disk" is in the keyword list the present invention then identifies the verb phrase "Analyze Disk." The process works exactly the same way for multiple word noun phrases inside the verb phrase (e.g., "Analyze Process Dump," instead of "Analyze Disk").

The lexicon 52 can also include single word regular expressions. If a regular expression is in the lexicon 52, then the natural language module 32 will identify any word in the input text that matches against the regular expression. Being able to define regular expressions in the lexicon 52 gives the maintainer of the lexicon 52 more flexibility than being restricted to defining literal words and phrases. For example, the term "SYS$:n+" can be defined to match all the VMS (an operating system available from Digital Equipment Corporation) operating system service routines instead of having to enter the name of every operating system service routine directly into the lexicon 52.

Some of the syntax rules of the singular expressions allowed in the lexicon 52 are that an ordinary character matches that character; a period matches any character; a colon matches a class of characters described by the following character, e.g., ":a" matches any alphabetic, ":d" matches digits, ":n" matches alphanumerics; an expression followed by an asterisk matches zero or more occurrences of that expression e.g., "fo*" notches "f" "fo" "foo", etc. and an expression followed by a plus sign matches one or more occurrences of that expression, e.g., "fo+" matches "foo, etc."

The output of the natural language module 32 is a list which is a collection of sublists where each sublist corresponds to a single sentence in the input text and contains all the recognized keywords in that sentence. This list is passed to the intelligent inferencer module 34 for further analysis and possible augmentation as is described below.

The intelligent inferencer module 34 takes the information extracted directly from the input text by the natural language module 32 and attempts to add to that information by deducing further facts that are implied by the keywords identified. This module 34 uses the keyword class hierarchy 54. Each class in the keyword class hierarchy 54 contains a group of keywords (already defined in the lexicon 52) that share something in common. The classes are structured into a hierarchy such that classes themselves can be members of other classes. An exemplary portion of the keyword class hierarchy 54 is illustrated in FIG. 4.

What is useful about these classes is that facts can be attached to them to deduce implied information if a member of a class is found in the input text. If a keyword class member is identified, then all the facts attached to that class are inferred and added to the list of deduced facts. In addition, all the facts attached to the parent classes are inferred and added to the list of deduced facts as well.

In addition to inferring new facts with keyword classes, more general descriptions of an identified keyword can be substituted in an attempt to match other key phrases. This process is called "keyword substitution." It is an attempt to match key phrases in the lexicon 52 that could not be matched explicitly. For example, it may be desirable to match the phrase "Analyze Disk" every time "Analyze X" is detected, where X is a specific disk device. This is accomplished without having to enter a single verb phrase for every specific disk device into the lexicon 52 which would cause the maintenance of a lexicon to become problematic.

Using keyword substitution, a group of like devices can be grouped into a class and a word attached to the class to be used as a substitute for matching phrases in the lexicon 52. Going back to the example above, a class of disk devices can be defined and the keyword "disk" can be associated as a substitute. This way, "Analyze RD54" (where RD54 is a model number of a disk drive) can be recognized as "Analyze Disk" without having to have "Analyze RD54" stored in the lexicon 52.

The output of the intelligent inferencer module 34 is the list of all the extracted keywords and the list of all the deduced facts that the intelligent inferencer module 34 was able to infer. Associated with each extracted keyword is a number designating the frequency of the keyword in the input text.

An exemplary embodiment of the modules which comprise the intelligent inferencer module 34 are shown in FIG. 5. The left hand side of FIG. 5 shows the two main modules of the intelligent inferencer module 34, a fact inferencer module 60 and a keyword substitution module 62. The right hand side of FIG. 5 shows that both modules 60 and 62 use the keyword class hierarchy 54 (the same one illustrated in FIG. 3) as their knowledge base. The fact inferencer module 60 only utilizes the facts associated with the classes in the keyword class hierarchy 54 and the keyword substitution module 62 only uses the keyword substitutes associated with the classes in the keyword class hierarchy 54.

The fact inferencer module 60 follows a general method for attaching facts to keywords. This method, which is repeated for each keyword K, first searches the keyword class hierarchy 54 for all classes C, of which the identified keyword is a member. Then, all facts associated with C are added to a global list of deduced facts for each identified class C that K is a member. The step of adding all facts associated with the identified class C is then applied recursively on all of the parent classes of C. By following this method, the fact inferencer module 60 adds facts to the list of deduced facts.

The keyword substitution module 62 similarly follows a general method for substituting keywords. This method, which is repeated for each keyword K, first searches the keyword class hierarchy 54 for all classes C, of which K is a member. Then, all the substitution keywords S, associated with C are retrieved for each identified class C where K is a member. Then, S is substituted for K and an attempt is made to match verb phrases in the lexicon 52. If a match is found, it is added to a global list of identified keywords. Then, the steps of retrieving substitution keywords and substituting keywords are recursively applied on all of the parent classes of C.

The similarity measuring module 36 is responsible for returning a numeric similarity score for each category in the keyword/category profile 56. Each score indicates how similar a given category is to the recognized keywords extracted from the natural language input text. The similarity measuring module 36 uses the knowledge base of keyword/category profiles 56 to determine similarity scores for all of the categories defined. Each category in the keyword/category profiles 56 has its own profile containing the keywords that are relevant to that category. Once the input text is parsed by the natural language module 32 and the intelligent inferencer module 34, a list of all the keywords present in the input text, as well as the number of times they occur in the input text called term frequency, is assembled by the similarity measuring module 36. The category profile can be represented as a n-dimensional vector of the form C=(c1, c2, . . . , cn), where n equals the total number of possible keywords in the lexicon 52 and the individual elements "ci" represents the corresponding profile weight of keyword "i" in the category profile. The input text can also be represented as a n-dimensional vector of the form T=(t1, t2, . . . , tn), where n is as above and "ti" represents the corresponding weight of keyword "i" in the input text. Similarity between a category and an input text can then be measured as the inner product between these corresponding vectors, which is defined as:

$$Sim(C,T)=SUM(i=1,n)\ (ci * ti).$$

The size of n can vary depending on the size of the keyword lexicon 52.

The similarity measuring module 36 includes a method for efficiently computing the inner product similarity measure so that when n becomes large the similarity measures can still be quickly calculated. The method assumes that each keyword in the lexicon 52 has a corresponding vector of categories that it provides evidence for and a profile weight for each category. This information can be quickly computed from the category profile vectors described above. This is accomplished by first initializing all similarity scores for all categories to zero. Then, for each keyword i identified in the input text and for each category j in the category vector of the keyword i, the keyword weight of keyword i is multiplied by the profile weight of category j. Then, the resulting product is added to the similarity score for the category j.

The foregoing method insures that only the identified keywords and the categories they provide evidence for are being multiplied together. All the other portions of the inner products will equal zero anyway since the keyword weights will be zero (i.e., the keywords were not identified in the input text). The run time performance of this method is significantly better than performing a straight summation of the products of the vector elements because of the large number of elements equaling zero in the vectors.

Like keywords, categories can also be grouped into hierarchically structured classes. This feature allows a lexicon maintainer to define category class profiles as well as category profiles. The run time system of the present invention automatically translates category class profiles into individual category profiles and incorporates them into existing category profiles. Category classes are also useful when writing disambiguation rules. By having category classes, a single rule can operate on an entire class rather than writing individual rules for each category in a class.

The initial weights for category profiles and keyword weights for input texts are ascertained by formulae used by the similarity measuring module 36 that uses both term frequency and collection frequency as input. In text classification terms, collection frequency is the number of category profiles in which a specific keyword occurs. The profile weight calculation formula is as follows:

$$PW=\log(CAT/CF)$$

where CAT equals the total number of defined categories and CF equals the collection frequency of the given keyword (this formula uses only collection frequency). Note that as CF increases, the profile weight decreases. This makes sense because if a keyword provides evidence for a large number of categories then its profile weight should be lower than a keyword that provides evidence for a small number of categories.

The keyword weight calculation formula is as follows:

$$KW=(TF * \log(CAT/CF))/CKW$$

where CAT and CF are as above, TF equals the term frequency of the keyword in the input text, and CKW is the combined keyword weight and is calculated as follows:

$$CKW=SQUARE\_ROOT(SUM(i=1, n)\ (SQUARE(tf * \log(CAT/cf))))$$

where n is the total number of keywords found in the input text, "tfi" and "cfi" are the term and collection frequencies for one of the found keywords, and CAT is as previously defined.

Once similarity scores have been calculated for all categories, the similarity measuring module 36 applies a dynamic threshold to the list of categories. This threshold is a given tuneable offset from the similarity score of the most similar category. In other words, if N is the highest similarity score for the input text and M is the pre-defined threshold offset, then N−M is the threshold value. All categories whose similarity scores are below the threshold value are discarded and those above the threshold value are compiled into a list and passed to the next module, along with the list of recognized keywords and the list of deduced facts.

As described above, the foregoing results can be passed directly to the external application 24, to the relevance feedback learning module 40 or the category disambiguation module 38. If the information is passed to the optional category disambiguation module 38, it uses a rule base to select certain categories over other categories based on the list of recognized keywords and the list of deduced facts. Rules are utilized to decide the appropriate category when more than one category is a potential candidate for being the most similar. The left hand sides of the rules consist of CATEGORY and KEYWORD slot-value pairs and deduced facts. The right hand sides of the rule merely assert a preselected preference for one category over another category (or set of categories).

An example of a rule that could be used by the category disambiguation module 38 is set forth below.

```
IF:     (CATEGORY = VM-FILE-SYSTEM)
        (CATEGORY = ?X = (one-of DECNET-VAX
        VMS-TAPE))
        (DEVICE_TYPE = DISK)
        (KEYWORD = "ACP")
THEN:   PREFER VMS-FILE-SYSTEM OVER ?X
```

This rule states that if VMS-FILE-SYSTEM and either DECNET-VAX or VMS-TAPE are potentially most similar categories, and if the fact (DEVICE_TYPE=DISK) was deduced by the intelligent inferencer module 34, and if the keyword ACP has been found (or one of its synonyms); then the category VMS-FILE-SYSTEM will be preferred over either DECNET-VAX or VMS-TAPE.

When the category disambiguation module 38 is invoked, all the rules that can apply to the given input text are fired and all the category preferences are recorded by the category disambiguation module 38. As a result of the firing of the rules, the list of categories whose similarity scores are above the threshold value is modified to include only the most similar categories that do not have any other category with preference over them. This list, along with the list of recognized keywords and the list of deduced facts, is then passed to the application 24 (and to the relevance feedback learning module 40).

As described above, the category disambiguation module 38 is detachable from the run time architecture of the present invention. If a particular text classification application has no heuristics for category selection, then the category disambiguation module 38 can be bypassed and reliance can be placed solely on similarity scores calculated by the similarity measuring module 36, to determine the most similar category. Detaching the rule base will most likely result in a decrease in the accuracy of the classification; but for some applications no such rule base exists. By making the rule base detachable, the range of potential applications that can be developed using the present invention is increased.

An exemplary implementation of the category disambiguation module 38 is illustrated in FIG. 6. The top portion of FIG. 6 shows the compile time processing needed to translate the category selection rule base 58 into the proper syntax for a run time inference engine, such as "CLIPS," which is a public domain inference engine developed by NASA. A rule compiler 68 takes as input a category selection rule base 64 and category class hierarchy 66. At run time, all the recognized keywords, facts, and most similar categories (that are given as input to the category disambiguation module) are translated into CLIPS facts 72 and are given as input (along with a CLIPS rule base 70) to the CLIPS inference engine 74. The CLIPS inference engine 74 fires as many rules as it can against the given facts. Each rule firing returns a category preference. Once all the rules that can fire have fired, then all the category preferences are collected and used by the present invention to come up with a final list of most similar categories (as described above).

Once text classification is done and the information is passed to the external application 24 for further application specific processing, the optional relevance feedback learning module 40 can be invoked to adjust the keyword/category profile weights to achieve better accuracy. The module 40 collects all the text classifications over a predetermined period from either the similarity measuring module 36 or the optional category disambiguation module 38, whichever is the last module of the run time system. The classifications include the input text, the chosen most similar category, and the keyword weights for the extracted keywords. Then, the relevance feedback learning module 40 performs the following tasks for each category profile in the keyword/category profiles 56. First, all the text classifications where the particular category was identified as the most similar are collected. Next, the input texts which were correctly classified and which were not are determined. Then, the keyword weights for all the correctly classified input texts are added to the corresponding keyword profile weights in the category profile. Finally, the keyword weights for all the incorrectly classified input texts are subtracted from the corresponding keyword profile weights in the category profile. Also, the correct category is determined and the keyword weights are added to the profile of that category. An example of an application that may use the text classification system of the present invention is routing of customer service requests within a customer support center. Without an automated text classifier, human call screeners interact with a call handling system and determine the appropriate group to send a customer service request. A call handling system records all the pertinent information that a support specialist needs to solve the customer problem. With an automated text classifier, the call handling system can automatically invoke the text classification system of the present invention to determine where to route the customer service request without human intervention.

The following section provides an example of a call handling application for a given customer service request and shows how the individual modules of the text classification system of the present invention operate on the customer service request. The output of the text classification system enables the application to route the customer service request to the appropriate group. Although not shown here because it is application specific processing, the call handling system would take this output and automatically send the customer service request to the identified support group.

Set forth below is an explanation of the processing performed by the system of the present invention using an example natural language text input. The example text input is:

"While trying to backup my database to a TK70, the process died with the error AIJDISABLED and produced a dump file. I need help analyzing the dump file and getting the backup to work."

This input text is passed to the run time system by the external application 24 in machine readable form. The following explanation demonstrates how the present invention processes this input text.

As discussed above, the processing begins with the natural language module 32. The natural language module 32 utilizes the lexicon 52 to recognize words or phrases in the natural language input text. Set forth below is an example of entries in the lexicon 52. Each entry in the lexicon 52 has a corresponding identifier which defines the entry type. For example, "BACKUP" is identified as a verb and a noun in separate entries.

BACKUP	VERB
BACKUP	NOUN
DATABASE	NOUN
AIJDISABLED	NOUN
DUMP FILE	NOUNPHRASE
ANALYZE DUMP	VERB PHRASE
TK:D:D+	REGULAR EXPRESSION

Given the entries in the lexicon 52, the natural language module 32 identifies the following keywords and phrases from the given natural language input text:

backup (twice, once as a verb and once as a noun), database, AIJDISABLED, TK70 (matched against the regular expression), dump file, and analyze dump. Two interesting events happen in the recognition of the verb phrase, "analyze dump." The first is that a morphological variant of the verb "analyze" is identified ("analyzing" being the morphological variant). The second is that the phrase was recognized as a single unit even though the two words that comprise it were not contiguous in the input text. As previously described, the natural language module 32 identifies the verb portion of the verb phrase and then looks for an occurrence of the noun portion in the same sentence. In this case it was successful, so the entire verb phrase matches. The list that the natural language module 32 outputs as a data structure to the intelligent inferencer module 34 as the result of parsing this input text would look something like this:

```
((S1:      ("BACKUP" 1)  ("DATABASE" 1)
           ("TK-DEVICE" 1)
           ("AIJDISABLED" 1)  ("DUMP FILE" 1))
((S2:      ("ANALYZE DUMP" 1)  ("BACKUP" 1)).
```

The numbers with each identified keyword represent the frequency of the keyword in the given sentence. This data structure is passed as input to the intelligent inferencer module 34.

The intelligent inferencer module 34 uses class information to deduce further information from the input text. For the purposes of this example, the following classes are defined to reside in the keyword class hierarchy 54:

Keyword Class TAPE-DEVICES, which is a grouping of all specific tape devices and includes "TK70",

Keyword Class RDB-ERROR-MSGS, which is a grouping of all the error messages generated by the product RDB, and includes "AIJDISABLED",

Keyword Class ERROR-MSGS, which is a grouping of all possible error messages and includes the keyword class RDB-ERROR-MSGS,

Category Class VIA-PRODUCTS, which is a grouping of all the VIA products, including RDB (RDB is a category in the domain-specific knowledge base).

The following facts are associated with the above classes in the keyword class hierarchy 54:

(CLASS = TAPE-DEVICE)	→ (DEVICE-TYPE = TAPE),
(CLASS = RDB-ERROR-MSGS)	→ (LAYERED-PROD = RDB),
(CLASS = ERROR-MSGS)	→ (ERROR-MESSAGE = $keyword).

Given these classes and associated facts, it can be deduced that the DEVICE-TYPE is TAPE because of the identification of TK70. A potential layered product is RDB because of the identification of AIJDISABLED as a RDB error message. An error message found in this input text is AIJDISABLED.

The list of recognized keywords and the list of deduced facts output by the intelligent inferencer module 34 as a data structure would look something like this:

```
((KEYWORDS:   ("BACKUP" 2) ("DATABASE" 1)
              ("TK-DEVICE" 1) ("AIJDISABLED" 1)
              ("DUMP FILE" 1) ("ANALYZE DUMP"
              1))
(FACTS:  (DEVICE-TYPE: TAPE) (LAYERED-PROD: RDB)
         (ERROR-MESSAGE: AIJDISABLED))).
```

Notice that the keywords are now not separated into sentence groupings and that the information deduced by the intelligent inferencer module 34 is incorporated into the data structure. This data structure is passed as input to the similarity measuring module 36.

The similarity measuring module 36 calculates a similarity measure for every category in the keyword/category profiles 56 against the identified keywords in the data structure. At this point, the frequency numbers associated with each keyword will be replaced by their term weights by using the term weighing formulae previously described. To keep things simple in this example, it is assumed that the term weights remain as they are above. For this example, the following category profiles contained in the keyword/category profiles 56:

Category BACKUP	has the associated keyword "BACKUP"
Category RDB	has the associated keywords "DATABASE" and "AIJDISABLED"
Category DBMS	has the associated keyword "DATABASE"
Category TAPE	has the associated keyword "TK-DEVICE"
Category BUGCHECK	has the associated keywords "DUMP FILE" and "ANALYZE DUMP"

It is assumed for this example that each keyword for each category has a weight of 1. It is also assumed that there are other categories in the knowledge base, but that none of them have any keywords in their profiles that match the keywords found in the input text. Also, it should be understood that the categories above have other keywords in their profiles, but for simplicity, only the keywords that match keywords found in the input text are presented. The similarity measures for the categories above are then as follows:

Sim(T, BACKUP) =	2 (because "BACKUP" has a keyword weight of 2)
Sim(T, RDB) = 2	
Sim(T, DBMS) = 1	
Sim(T, TAPE) = 1	
Sim(T, BUGCHECK) = 2	

For this example, a category threshold offset of 0.5 is chosen. This means that only the categories with similarity measures above 1.5 (2−0.5) will pass on to the next module. The list of the most similar categories, along with the list of recognized keywords and the list of deduced facts, that the similarity measuring module 36 outputs as a data structure would look something like this:

```
((KEYWORDS:   ("BACKUP" 2) ("DATABASE" 1)
              ("TK-DEVICE" 1) ("AIJDISABLED" 1)
              ("DUMP FILE" 1) ("ANALYZE DUMP"
              1))
(FACTS:       (DEVICE-TYPE: TAPE) (LAYERED-
```

-continued

```
                PROD: RDB) (ERROR-MESSAGE:
                AUDISABLED))
(CATEGORIES:    (BACKUP 2) (RDB 2) (BUGCHECK 2))).
```

To continue the example, a rule base is selected. There are two rules in the category selection rule base 58 as follows:

```
IF  (KEYWORD = "BACKUP") and
    (LAYERED-PROD = VIA-PRODUCTS)
THEN (PREFER VIA-PRODUCTS OVER BACKUP)
IF  (SKILL = BUGCHECK) and
    (LAYERED-PROD = VIA-PRODUCTS)
and
      (NOT (EXISTS BUGCHECK-TYPE))
THEN (PREFER VIA-PRODUCTS OVER BUGCHECK)
```

These two rules use the class VIA-PRODUCTS which we defined previously as including the category RDB. Since the fact (LAYERED-PROD=RDB) is present, both of these rules will fire, the result being that the RDB category is preferred over both BACKUP and BUGCHECK. The final data structure output by the category disambiguation module 38, is as follows:

```
((KEYWORDS:      ("BACKUP" 2) ("DATABASE" 1)
                 ("TK-DEVICE" 1) ("AUDISABLED" 1)
                 ("DUMP FILE" 1) ("ANALYZE DUMP"
                 1))
(FACTS:          (DEVICE-TYPE: TAPE) (LAYERED-
                 PROD: RDB) (ERROR-MESSAGE:
                 AUDISABLED))
(CATEGORIES:     (RDB 2))
(PREFERENCES:    (RDB OVER BACKUP RULE-1) (RDB
                 OVER BUGCHECK RULE-2))).
```

The rule numbers are listed with the preferences so they can be accessed at run time to generate explanations to the user as to why the rules fired.

Once a text classification operation is performed, control is returned to the invoking application, in this case a call handling system. The call handling system will then use the classification to route the customer service request to the appropriate support group. The call handling system can also store the service request and its classification for later use by the relevance feedback learning module 40 (FIG. 2). After a given predetermined period of time, the call handling system collects all the service requests and their classifications and passes them as input to the relevance feedback learning module 40. The relevance feedback learning module 40 takes these classified requests and interacts with a human call routing expert via video display terminal 18 (FIG. 1) to determine which ones were correctly and which ones were incorrectly routed. This learning module would then take the information from the call routing expert and adjust the profile weights in the keyword/category profiles 56 as previously described.

What is claimed is:

1. A method for classifying natural language text input into a computer system, the system includes memory having a domain specific knowledge base having a plurality of categories stored therein, the method comprising the steps of:

(a) accepting as input natural language input text;

(b) parsing the natural language input text into a first list of recognized keywords;

(c) using the first list to deduce further facts from the natural language input text;

(d) compiling the deduced facts into a second list;

(e) calculating a numeric similarity score for each one of the plurality of categories in the knowledge base to indicate how similar one of the plurality of categories is to the natural language input text;

(f) applying a dynamic threshold to determine which ones of the plurality of categories are most similar to the recognized keywords of the first list, comprising the sub-steps of:

(I) calculating a value for the dynamic threshold based upon a similarity score of a most similar category and a predefined threshold offset, and

(II) classifying the categories based upon their respective similarity scores by discarding categories whose similarity scores are below the threshold value;

(g) compiling the ones of the plurality of categories determined to be most similar in step (f) into a third list; and

(i) passing the first list, the second list and the third list to an external application.

2. The method according to claim 1 wherein the keywords comprise words, phrases and regular expressions.

3. The method according to claim 1 wherein the knowledge base includes a keyword class hierarchy structured such that keywords that share something in common are grouped into classes, each class has associated facts that are true when a member of the class is identified in the natural language input text, wherein the steps of using the first list to deduce further facts from the natural language input text and compiling the deduced facts into a second list further are performed by the steps of:

(a) searching the keyword class hierarchy to determine if a keyword identified in the first list is a member of a class in the keyword class hierarchy;

(b) when a keyword identified in the first list is a member of a class,

(i) inferring all the facts attached to that class by adding them to the second list, and

(ii) adding all the facts attached to all classes above the classes of which the identified keyword is a member in the keyword class hierarchy to the second list; and

(c) repeating steps (a) through (b) for each keyword in the first list.

4. The method according to claim 2 wherein the knowledge base includes a keyword class hierarchy structured such that keywords that share something in common are grouped into classes, each class has associated facts that are true when a member of the class is identified in the natural language input text, wherein the step of using the first list to deduce further facts from the natural language input text further comprises the step of substituting general descriptions of an identified keyword in the first list in an attempt to match other phrases that could not be matched explicitly so that a group of similar keywords can be grouped into a class and a word can be attached to the class to be used as a substitute for matching phrases.

5. The method according to claim 1 wherein the knowledge base includes a keyword class hierarchy structured such that keywords that share something in common are grouped into classes, each class has associated facts that are true when a member of the class is

identified in the natural language input text, wherein the steps of using the first list to deduce further facts from the natural language input text and compiling the deduced facts into a second list further are performed by the steps of:

(a) searching the keyword class hierarchy for all classes of which an identified keyword in the first list is a member;

(b) adding all facts associated with each one of the classes of which the identified keyword is a member to a global list of deduced facts;

(c) recursively applying step (b) on all classes above the classes of which the identified keyword is a member in the keyword class hierarchy; and

(d) repeating steps (a) through (c) for each keyword in the first list.

6. The method according to claim 1 wherein the knowledge base includes a lexicon that includes words, phrases and expressions, and a keyword class hierarchy structured such that keywords that share something in common are grouped into classes, each class has associated facts that are true when a member of the class is identified in the natural language input text, wherein the step of using the first list to deduce further facts from the natural language input text further comprises the steps of:

(a) searching the keyword class hierarchy for all classes of which an identified keyword in the first list is a member;

(b) locating all substitution keywords associated with each class of which the identified keyword is a member;

(c) retrieving the located substitution keywords;

(d) substituting the located substitution keywords for the identified keyword;

(e) using the located substitution keywords to identify matches between the located substitution keywords and phrases in the lexicon;

(f) recursively applying steps (b) through (e) on all classes above the classes of which the identified keyword is a member in the keyword class hierarchy; and

(g) repeating steps (a) through (f) for each keyword in the first list.

7. A text classification system comprising:

memory;

a domain specific knowledge base stored in said memory having a plurality of categories, the domain specific knowledge base includes a knowledge base of keyword/category profiles, each category in the keyword/category profiles knowledge base having an associated profile which indicates what information provides evidence for a given category, the keyword/profile weight knowledge base arranged to have associated with each keyword in a profile a profile weight that represents the amount of evidence a keyword provides for a given category; and

a computer coupled to the memory, the computer including:

a natural language module for accepting as input into the computer natural language input text, the natural language module includes means for parsing the natural language input text into a first list of recognized keywords;

an intelligent inferencer module for using the first list to deduce further facts from the information explicitly stated in the natural language input text, the intelligent inferencer module includes means for compiling the deduced facts into a second list;

a similarity measuring module for calculating a numeric similarity score for each one of the plurality of categories in the knowledge base to indicate how similar one of the plurality of categories is to the natural language input text, the similarity measuring module includes:

means for applying a dynamic threshold to determine which ones of the plurality of categories are most similar to the recognized keywords of the natural language input text, and

means for compiling the ones of the plurality of categories determined to be most similar into a third list; and

a relevance feedback learning module for adjusting the profile weights in the keyword/category profiles in the domain specific knowledge base based upon the ones of the plurality of categories determined most relevant to the natural language input text by the similarity measuring module and a second ones of the plurality of categories determined most relevant to the natural language input text by an external source.

8. A method for classifying natural language text input into a computer system, the system includes memory having a domain specific knowledge base having a plurality of categories stored therein, the method comprising the steps of:

(a) accepting as input natural language input text;

(b) parsing the natural language input text into a first list of recognized keywords;

(c) using the first list to deduce further facts from the natural language input text;

(d) compiling the deduced facts into a second list;

(e) calculating a numeric similarity score for each one of the plurality of categories in the knowledge base to indicate how similar one of the plurality of categories is to the natural language input text;

(f) applying a dynamic threshold to determine which ones of the plurality of categories are most similar to the recognized keywords of the first list, the step of applying a dynamic threshold further comprising the sub-steps of:

(1) calculating a value for the dynamic threshold based upon a similarity score of a most similar category and a predefined threshold offset, and

(2) classifying the categories based upon their respective similarity scores by discarding categories whose similarity scores are below the threshold value; and

(g) compiling the ones of the plurality of categories determined to be most similar in step (f) into a third list.

9. The method according to claim 1 wherein the domain specific knowledge base further includes a rule base, the method further comprising the steps of:

(a) utilizing the rule base to select certain ones of the plurality of categories determined to be most similar to the recognized keywords over other ones of the plurality of categories based on the first and second lists; and

(b) modifying the third list of the most similar categories to include the certain ones of the plurality of categories selected.

10. The method according to claim 1 wherein the domain specific knowledge base includes a knowledge

base of keyword/category profiles, each category in the keyword/category profiles knowledge base having an associated profile which indicates what information provides evidence for a given category, the keyword/profile weight knowledge base is arranged to have associated with each keyword in a profile a profile weight that represents the amount of evidence a keyword provides for a given category, the method further comprising the step of adjusting the profile weights in the keyword/category profiles in the domain specific knowledge base based upon the ones of the plurality of categories determined most relevant to the natural language input text and a second ones of the plurality of categories determined most relevant to the natural language input text by an external source.

11. A method for routing customer service requests by a computer system in a customer support center which includes support groups to service customer requests, the computer system including a call handling system, a text classification system and memory having a domain specific knowledge base having a plurality of categories stored therein representative of the support groups within the customer support center, each support group being identified by a name, the method comprising the steps of:

(a) receiving a customer service request by the computer system from the call handling system;
(b) passing the customer service request to the text classification system to determine where to route the customer service request within the customer support center;
(c) parsing the customer service request into a first list of recognized keywords;
(d) using the first list to deduce further facts from the customer service request;
(e) compiling the deduced facts into a second list;
(f) calculating a numeric similarity score for each one of the plurality of categories in the knowledge base to indicate how similar each one of the plurality of categories is to the the customer service request;
(g) applying a dynamic threshold to identify which one of the support groups should handle the customer service request by determining which ones of the plurality of categories are most similar to the recognized keywords of the customer service request;
(h) compiling the ones of the plurality of categories determined to be most similar in step (g) into a third list;
(i) passing the first list, the second list and the third list back to the call handling system; and
(j) routing the customer service request to the identified one of the support groups.

12. A method for routing customer service requests by a computer system in a customer support center which includes support groups to service customer requests, the computer system including a call handling system, a text classification system and memory having a domain specific knowledge base having a plurality of categories stored therein representative of the support groups within the customer support center, each support group being identified by a name, and a rule base, the method comprising the steps of:

(a) receiving a customer service request by the computer system from the call handling system;
(b) passing the customer service request to the text classification system to determine where to route

the customer service request within the customer support center;
(c) parsing the customer service request into a first list of recognized keywords;
(d) using the first list to deduce further facts from the customer service request;
(e) compiling the deduced facts into a second list;
(f) calculating, utilizing the first list, a numeric similarity score for each one of the plurality of categories in the knowledge base to indicate how similar each one of the plurality of categories is to the customer service request;
(g) applying a dynamic threshold to identify which support groups should handle the customer service request by determining which ones of the plurality of categories are most similar to the recognized keywords of the customer service request;
(h) compiling the ones of the plurality of categories determined to be most similar in step (g) into a third list;
(i) utilizing the rule base to select certain ones of the plurality of categories determined to be most similar to the recognized keywords over other ones of the plurality of categories based on the first and second lists;
(j) modifying the third list of the most similar categories to include the certain ones of the plurality of categories selected;
(k) passing the first list, the second list and the third list back to the call handling system; and
(l) routing the customer service request to the selected one of the support groups.

13. The method according to claim **11** or **12** wherein the domain specific knowledge base includes a knowledge base of keyword/category profiles, each category in the keyword/category profiles knowledge base having an associated profile which indicates what information provides evidence for a given category, the keyword/profile weight knowledge base is arranged to have associated with each keyword in a profile a profile weight that represents the amount of evidence a keyword provides for a given category, the method further comprising the step of adjusting the profile weights in the keyword/category profiles in the domain specific knowledge base based upon the one of the support groups selected to handle the customer service request and a second one of the support groups determined most relevant to the natural language input text by an external source.

14. A text classification system comprising:
a memory;
a domain specific knowledge base stored in said memory having a plurality of categories wherein the domain specific knowledge base includes a knowledge base of keyword/category profiles, each category in the keyword/category profiles knowledge base having an associated profile which indicates what information provides evidence for a given category, the keyword/profile knowledge base is arranged to have associated with each keyword in a profile a profile weight that represents the amount of evidence a keyword provides for a given category; and
a computer coupled to the memory, the computer including:
means for accepting as input into the computer, natural language input text,

means for parsing the natural language input text into a first list of recognized keywords,

means for using the first list to deduce further facts from the natural language input text,

means for compiling the deduced facts into a second list,

means for calculating a numeric similarity score for each one of the plurality of categories in the knowledge base to indicate how similar one of the plurality of categories is to the natural language input text,

means for applying a dynamic threshold to determine which ones of the plurality of categories are most similar to the recognized keywords of the first list,

means for adjusting the profile weights in the keyword/categories determined to be the most relevant to the natural language input text and a second ones of the plurality of categories determined most relevant to the natural language input text by an external source,

means for compiling the ones of the plurality of categories determined to be most similar into a third list, and

means for passing the first list, the second list and the third list to an external application.

15. The text classification system according to claim **14** wherein the keywords comprises words, phrases and regular expressions.

16. The text classification system according to claim **14** wherein the domain specific knowledge base further includes a rule base and the computer further comprises:

means for utilizing the rule base to select certain ones of the plurality of categories that were determined to be most similar to the recognized keywords over other ones of the plurality of categories based on the first and second lists; and

means for modifying the third list of the most similar categories to include the certain ones of the plurality of categories selected.

17. The text classification system according to claim **14** wherein the domain specific knowledge base includes a knowledge base of keyword/category profiles, each category in the keyword/category profiles knowledge base having an associated profile which indicates what information provides evidence for a given category, the keyword/profile weight knowledge base is arranged to have associated with each keyword in a profile a profile weight that represents the amount of evidence a keyword provides for a given category, wherein the computer further comprises means for adjusting the profile weights in the keyword/category profiles in the domain specific knowledge base based upon the ones of the plurality of categories determined most relevant to the natural language input text and a second ones of the plurality of categories determined most relevant to the natural language input text by an external source.

18. A method for classifying natural language text input into a computer system, the system includes memory having a domain specific knowledge base having a plurality of categories stored therein and including a rule base, the method comprising the steps of:

(a) accepting as input natural language input text;

(b) parsing the natural language input text into a first list of recognized keywords;

(c) using the first list to deduce further facts from the natural language input text;

(d) compiling the deduced facts into a second list;

(e) calculating a numeric similarity score for each one of the plurality of categories in the knowledge base to indicate how similar one of the plurality of categories is to the natural language input text;

(f) applying a dynamic threshold to determine which ones of the plurality of categories are most similar to the recognized keywords of the first list;

(g) compiling the ones of the plurality of categories determined to be most similar in step (f) into a third list;

(h) utilizing the rule base to select certain ones of the plurality of categories determined to be most similar to the recognized keywords over other ones of the plurality of categories based on the first and second lists; and

(i) modifying the third list of the most similar categories to include the certain ones of the plurality of categories selected.

19. The text classification system according to claim **14** wherein the means for applying a dynamic threshold further includes:

means for calculating a value for the dynamic threshold based upon a similarity score of a most similar category and a predefined threshold offset; and

means for classifying the categories based upon their respective similarity scores by discarding categories whose similarity scores are below the threshold value.

20. A method for classifying natural language text input into a computer system, the system includes memory having a domain specific knowledge base having a plurality of categories stored therein, the knowledge base including a lexicon that includes words, phrases and expressions and a keyword class hierarchy structured such that keywords that share something in common are grouped into classes, each class has associated facts that are true when a member of the class is identified in the natural language inputs text, the method comprising the steps of:

(a) accepting as input natural language input text;

(b) parsing the natural language input text into a first list of recognized keywords;

(c) using the first list to deduce further facts from the natural language input text comprising the sub-steps of:

(1) searching the keyword class hierarchy for all classes of which an identified keyword in the first list is a member,

(2) locating all substitution keywords associated with each class of which the identified keyword is a member,

(3) retrieving the located substitution keywords,

(4) substituting the located substitution keywords for the identified keyword,

(5) using the located substitution keywords to identify matches between the located substitution keywords and phrases in the lexicon,

(6) recursively applying sub-steps (2) through (5) on all classes above the classes of which the identified keyword is a member in the keyword class hierarchy, and

(7) repeating sub-steps (1) through (6) for each keyword in the first list;

(d) compiling the deduced facts into a second list;

(e) calculating a numeric similarity score for each one of the plurality of categories in the knowledge base to indicate how similar one of the plurality of categories is to the natural language input text;

(f) applying a dynamic threshold to determine which ones of the plurality of categories are most similar to the recognized keywords of the first list; and

(g) compiling the ones of the plurality of categories determined to be most similar in step (f) into a third list.

21. A text classification system comprising:

memory;

a domain specific knowledge base stored in said memory having a plurality of categories, the domain specific knowledge base including a rule base; and

a computer coupled to the memory, the computer including:

a natural language module for accepting as input into the computer natural language input text, the natural language module includes means for parsing the natural language input text into a first list of recognized keywords;

an intelligent inferencer module for using the first list to deduce further facts from the information explicitly stated in the natural language input text, the intelligent inferencer module includes means for compiling the deduced facts into a second list;

a similarity measuring module for calculating a numeric similarity score for each one of the plurality of categories in the knowledge base to indicate how similar one of the plurality of categories is to the natural language input text, the similarity measuring module includes:

means for applying a dynamic threshold to determine which ones of the plurality of categories are most similar to the recognized keywords of the natural language input text, and

means for compiling the ones of the plurality of categories determined to be most similar into a third list; and

a category disambiguation module for utilizing the rule base to select certain ones of the plurality of categories determined to be most similar to the recognized keywords over other ones of the plurality of categories based on the first and second lists, the category disambiguation module includes means for modifying the third list of the most similar categories to include the certain ones of the plurality of categories selected.

22. A text classification system comprising:

a memory;

a domain specific knowledge base stored in said memory having a rule base and a plurality of categories; and

a computer coupled to the memory, the computer including:

means for accepting as input into the computer, natural language input text,

means for parsing the natural language input text into a first list of recognized keywords,

means for using the first list to deduce further facts from the natural language input text,

means for compiling the deduced facts into a second list,

means for calculating a numeric similarity score for each one of the plurality of categories in the knowledge base to indicate how similar one of

the plurality of categories is to the natural language input text,

means for applying a dynamic threshold to determine which ones of the plurality of categories are most similar to the recognized keywords of the first list,

means for compiling the ones of the plurality of categories determined to be most similar into a third list,

means for utilizing the rule base to select certain ones of the plurality of categories that were determined to be most similar to the recognized keywords over other ones of the plurality of categories based on the first and second lists, and

means for modifying the third list of the most similar categories to include the certain ones of the plurality of categories selected.

23. A text classification system comprising:

a memory;

a domain specific knowledge base stored in said memory having a plurality of categories; and

a computer coupled to the memory, the computer including:

means for accepting as input into the computer, natural language input text,

means for parsing the natural language input text into a first list of recognized keywords,

means for using the first list to deduce further facts from the natural language input text,

means for compiling the deduced facts into a second list,

means for calculating a numeric similarity score for each one of the plurality of categories in the knowledge base to indicate how similar one of the plurality of categories is to the natural language input text,

means for applying a dynamic threshold to determine which ones of the plurality of categories are most similar to the recognized keywords of the first list,

means for calculating a value for the dynamic threshold based upon a similarity score of a most similar category and a predefined threshold offset,

means for classifying the categories based upon their respective similarity scores by discarding categories whose similarity scores are below the threshold value, and

means for compiling the ones of the plurality of categories determined to be most similar into a third list.

24. A text classification system comprising:

a memory;

a domain specific knowledge base stored in said memory having a plurality of categories, the domain specific knowledge base including a knowledge base of keyword/category profiles, each category in the keyword/category profiles knowledge base having an associated profile which indicates what information provides evidence for a given category, the keyword/profile weight knowledge base is arranged to have associated with each keyword in a profile a profile weight that represents the amount of evidence a keyword provides for a given category; and

a computer coupled to the memory, the computer including:

means for accepting as input into the computer, natural language input text,

means for parsing the natural language input text into a first list of recognized keywords,

means for using the first list to deduce further facts from the natural language input text,

means for compiling the deduced facts into a second list,

means for calculating a numeric similarity score for each one of the plurality of categories in the knowledge base to indicate how similar one of the plurality of categories is to the natural language input text,

means for applying a dynamic threshold to determine which ones of the plurality of categories are most similar to the recognized keywords of the first list,

means for compiling the ones of the plurality of categories determined to be most similar into a third list, and

means for adjusting the profile weights in the keyword/category profiles in the domain specific knowledge base based upon the ones of the plurality of categories determined most relevant to the natural language input text and a second ones of the plurality of categories determined most relevant to the natural language input text by an external source.

* * * * *